Routing Problems in Order Picking

Schriftliche Promotionsleistung

zur Erlangung des akademischen Grades

Doctor rerum politicarum

vorgelegt und angenommen

an der Fakultät für Wirtschaftswissenschaft

der Otto-von-Guericke-Universität Magdeburg

Verfasser:	Dipl.-Wirt.-Math. André Scholz
Geburtsdatum und -ort:	12.07.1988 in Magdeburg
Arbeit eingereicht am:	28.04.2017

Gutachter der schriftlichen Promotionsleistung:

Prof. Dr. Gerhard Wäscher

apl. Prof. Dr. Andreas Bortfeldt

Datum der Disputation:	23.11.2017

Contents

Part I:

Outline of the Thesis

Outline of the Thesis

Order picking deals with the retrieval of requested items from their storage locations in the warehouse (Petersen & Schmenner, 1999; Wäscher, 2004). The items to be retrieved (picked) are specified by a set of external or internal customer orders. Although several attempts have been made to automate the picking process, manual order picking systems are prevalent in practice (de Koster et al., 2007). Due to the employment of human operators (order pickers) on a large scale in manual systems, order picking represents the most cost-intensive warehouse function, accounting for between 50% (Frazelle, 2002) and 65% (Coyle et al., 1996) of the total warehouse operating costs. Among manual systems, picker-to-part order picking systems are the most important ones (de Koster, 2008). In such systems, order pickers process the customer orders by performing tours through the picking area of the warehouse. Customer orders processed on the same tour are referred to as a picking order. Each tour starts and ends at the depot and includes all storage locations of the requested items (pick locations) contained in the respective picking order. The time spent for performing a tour can be divided into the time for preparing a tour, the time required at the pick locations for the identification and the retrieval of the items, and the time needed to travel from the depot to the first pick location, between the pick locations and from the last pick location back to the depot. From these components, the time for traveling represents the major part of an order picker's working time (Tompkins et al., 2010). Therefore, the minimization of the travel times of all tours (total travel time) is of prime importance for an efficient organization of the picking operations. Since the travel time is a linearly increasing function of the length of the corresponding tour (Jarvis & McDowell, 1991), the minimization of the lengths of all tours (total tour length) is equivalent to the minimization of the total travel time.

The length of a tour is dependent on the sequence according to which the pick locations included in the tour are meant to be visited. The determination of the sequence is part of the Picker Routing Problem which can be stated as follows. Let a set of picking orders consisting of requested items with known storage locations be given. For each picking order, the sequence according to which the pick locations are to be visited and the corresponding path through the picking area of the warehouse have then to be determined in such a way that the total tour length is minimized.

The Picker Routing Problem has been widely studied in the literature and a large variety of solution approaches exists. However, most approaches rely on the application of simple routing strategies which may result in very long tours (Roodbergen, 2001). For example, the tours constructed by means of the routing strategy most frequently used in practice leads to tours which are up to 48% longer than an optimal tour (Theys et al., 2010). Since the generation of such long tours can be expected to have a significant negative impact on the efficiency of the picking process, the approaches proposed in the literature so far cannot be seen as satisfactory. Therefore, in this thesis, several variants and extensions of the Picker Routing Problem are addressed and more promising solution approaches are presented. All solution approaches have been implemented and extensive numerical experiments have been conducted in order to evaluate the performance of the approaches. The solution approaches and the results of the experiments have been published in peer-reviewed journals and in a working paper series.

In Scholz & Wäscher (2017b), a comprehensive overview of state-of-the-art solution approaches to the Picker Routing Problem is given, while the approaches are classified according to the underlying assumptions. It is pointed out that the complexity of the Picker Routing Problem and the computational effort of corresponding solution approaches are mainly dependent on assumptions concerning the layout of the picking area, i.e. the arrangement of the storage locations in the picking area of the warehouse. The picking area includes picking aisles and cross aisles. Picking aisles have to be entered in order to retrieve items as the storage locations are situated on one side or even both sides of the picking aisles. Cross aisles do not contain any storage locations, but they enable the order pickers to switch between picking aisles. Based on the arrangement of the picking and cross aisles, a conventional or a non-conventional layout is constituted. Typically, the picking area is assumed to follow a conventional layout (Roodbergen, 2001) which is also assumed in the following parts of this thesis. In conventional layouts, picking aisles and cross aisles are straight, of equal length and width, and arranged parallel to each other, respectively. Furthermore, the cross aisles intersect the picking aisles at right angles and divide the picking area into blocks and the picking aisles into subaisles, where a subaisle is the part of a picking aisle which belongs to the same block (see Fig. 1). Consequently, a conventional layout with m picking aisles and $q+1$ cross aisles includes q blocks and $q \cdot m$ subaisles.

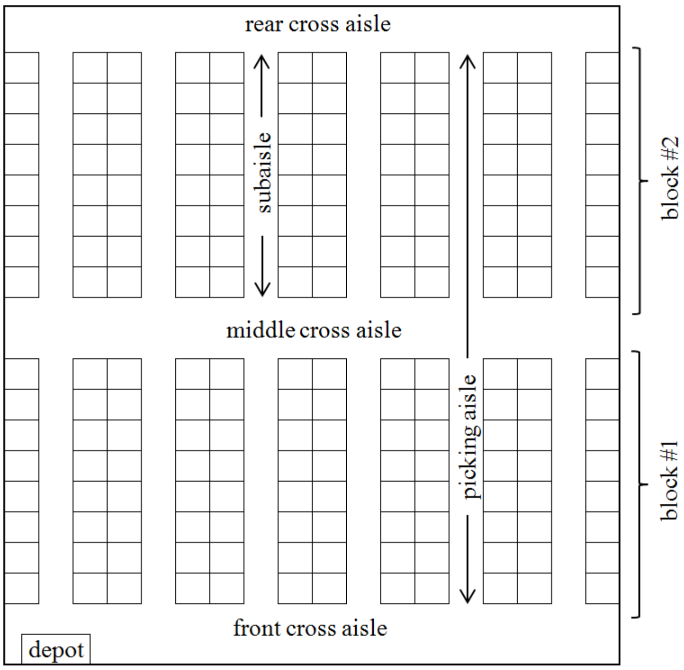


Fig. 1: Example of a conventional layout with two blocks

The layout of the warehouse is also characterized by the width of the picking aisles where standard, wide and narrow picking aisles can be distinguished. Standard picking aisles are wide enough such that order pickers can pass or overtake each other in such aisles. At the same time, standard picking aisles are narrow enough for allowing pickers to retrieve items from storage locations on both sides of the respective picking aisle without performing additional movements (Roodbergen, 2001). No such additional movements are required in narrow picking aisles. However, order pickers working at the same time in a narrow subaisle may cause congestion (Parikh & Meller, 2009). In contrast, congestion is not an issue when dealing with wide picking aisles but, in this case, additional movements are required for picking items located on different sides of the wide picking aisle (Goetschalckx & Ratliff, 1988).

Based on the characteristics of the layout, different solution approaches to the Picker Routing Problem are proposed in this thesis. First, conventional layouts with standard picking aisles are considered. If a conventional layout contains two cross aisles only, the picking area follows a so-called single-block layout. Using the special structure of optimal tours in a single-block layout (Ratliff & Rosenthal, 1983), a problem-specific model formulation has been developed by Scholz et al. (2016). The size of the model is independent of the number of pick locations and it only increases linearly with the number of picking aisles. By application of a commercial IP-solver to the model, any practical-sized problem instance can be solved to optimality within a small amount of computing time. However, when being adapted to conventional layouts with more than two cross aisles (multi-block layouts), the size of the model significantly increases. Therefore, several procedures have been applied to the underlying graph in Scholz (2017), reducing the size of the resulting model formulation. By means of numerical experiments, it has been demonstrated that the model formulation is suitable for solving Picker Routing Problems in multi-block layouts. In particular, the computing time does not increase with an increasing number of blocks, which can be seen as a major advantage of the model as no efficient solution approach exists which is able to deal with more than two blocks (Roodbergen, 2001).

In case of narrow picking aisles, order pickers are not able to pass or overtake each other, i.e. a picker may have to wait until another picker has performed the operations in a subaisle. Thus, the minimization of the total travel time does not represent a valid objective but rather waiting times have to be taken into account. In Hahn & Scholz (2017), problem parameters are first pointed out which have a significant impact on the waiting times of all order pickers (total waiting time) and situations are identified where the proportion of the total waiting time as part of the processing times of all customer orders is quite large. A truncated branch-and-bound algorithm is then proposed which aims for the minimization of the total waiting time. The results of the numerical experiments indicate that this approach provides high-quality solutions within short computing times.

In the above-mentioned parts of the thesis, picking orders are assumed to be given, i.e. decisions regarding the grouping (batching) of customer orders to picking orders have already been made. This type of decision is now integrated into the Picker Routing Problem, giving rise to the Joint Order Batching and Picker Routing Problem. The main characteristic of this problem can be found in the objective. Distance-related and tardiness-related objectives can be distinguished.

The minimization of the total tour length represents the most common distance-related objective. The benefit (in terms of the savings regarding the total tour length) of dealing with the Joint Order Batching and Picker Routing Problem instead of solving both subproblems in sequence has been investigated by Scholz & Wäscher (2017a). For this purpose, an iterated local search approach to the Order Batching Problem has been combined with several routing heuristics as well as with the exact approach of Roodbergen & de Koster (2001). By means of numerical experiments, it has been shown that the integration of the exact routing algorithm leads to superior results even if the computing time is limited to a few minutes.

If customer orders have to be completed until a certain due date, tardiness-related objectives are usually considered. The minimization of the tardiness of all customer orders (total tardiness), i.e. the extent to which the due dates are violated, represents a widely-used tardiness-related objective (Tsai et al., 2008). In contrast to the case of distance-related objectives, decisions regarding the assignment of picking orders to order pickers and the sequence according to which the picking orders are to be processed by the pickers have to be made as well. In Scholz et al. (2017), a variable neighborhood descent algorithm has been developed for solving this problem. The neighborhood structures are related to batching, assignment and sequencing decisions while two routing algorithms are used for the evaluation of the solutions. Numerical experiments have been conducted in order to show that the proposed algorithm is able to provide high-quality solutions within reasonable computing times. Furthermore, the benefit of dealing with all decisions simultaneously has been analyzed, and significant improvements compared to a sequential solution of the subproblems have been observed.

When customers place orders, the requested items have to be retrieved from their storage locations in the warehouse first. Problems arising in this context have been considered in the above-mentioned parts of the thesis. However, after having provided the items in the warehouse, vehicle tours have to be performed for shipping the requested items to the corresponding customer locations. In order to comply with the due dates of the customer orders, the picking and the shipping operations have to be well coordinated. This problem has been addressed by Schubert et al. (2017). In this paper, an iterated local search approach has been designed which contains neighborhood structures concerning the sequence of the picker tours as well as the composition and the sequence of the vehicle tours. Extensive numerical experiments have been executed in order to identify the situations where a holistic consideration of the picking and the shipping operations is inevitable and to point out in which cases both types of operations can be dealt with separately.

The thesis concludes with an outlook where several interesting areas for further research are identified based on the findings from this thesis.

References

Coyle, J. J.; Bardi, E. J. & Langley, C. J. (1996): The Management of Business Logistics. 6th ed., West Publishing Company: St. Paul.

de Koster, R.; Le-Duc, T. & Roodbergen, K. J. (2007): Design and Control of Warehouse Order Picking: A Literature Review. *Science Direct* 182, 481-501.

de Koster (2008): Warehouse Assessment in a Single Tour. *Facility Logistics: Approaches and Solutions to Next Generation Challenges*, Lahmar, M. (ed.), 39-60, Taylor & Francis Group: New York.

Frazelle, E. (2002): World-Class Warehouse and Material Handling. McGrawHill: New York.

Goetschalckx, M. & Ratliff, H. D. (1988): Order Picking in an Aisle. *IIE Transactions* 20, 53-62.

Hahn, S. & Scholz, A. (2017): Order Picking in Narrow-Aisle Warehouses: A Fast Approach to Minimize Waiting Times. Working Paper No. 6/2017, Faculty of Economics and Management, Otto-von-Guericke University Magdeburg.

Jarvis, J. M. & McDowell, E. D. (1991): Optimal Product Layout in an Order Picking Warehouse. *IIE Transactions* 23, 93-102.

Parikh, P. J. & Meller, R. D. (2009): Estimating Picker Blocking in Wide-Aisle Order Picking Systems. *IIE Transactions* 41, 232-246.

Petersen, C. G. & Schmenner, R. W. (1999): An Evaluation of Routing and Volume-Based Storage Policies in an Order Picking Operation. *Decision Science*, 30, 481-501.

Ratliff, H. D. & Rosenthal, A. R. (1983): Order-Picking in a Rectangular Warehouse: A Solvable Case of the Traveling Salesman Problem. *Operations Research* 31, 507-521.

Roodbergen, K. J. (2001): Layout and Routing Methods for Warehouses. Trial: Rotterdam.

Roodbergen, K. J. & de Koster, R. (2001a): Routing Order Pickers in a Warehouse with a Middle Aisle. *European Journal of Operational Research* 133, 32-43.

Scholz, A. (2016): An Exact Solution Approach to the Single-Picker Routing Problem in Warehouses with an Arbitrary Block Layout.*

Scholz, A.; Henn, S.; Stuhlmann, M. & Wäscher, G. (2016): A New Mathematical Programming Formulation for the Single-Picker Routing Problem. *European Journal of Operational Research* 253, 68-84.*

Scholz, A.; Schubert, D. & Wäscher, G. (2017): Order Picking with Multiple Pickers and Due Dates – Simultaneous Solution of Order Batching, Batch Assignment and Sequencing, and Picker Routing Problems. *European Journal of Operational Research* 263, 461-478.*

Scholz, A. & Wäscher, G. (2017a): Order Batching and Picker Routing in Manual Order Picking Systems: The Benefits of Integrated Routing. *Central European Journal of Operations Research* 25, 491-520.*

Scholz, A. & Wäscher, G. (2017b): Picker Routing in Manual Picker-to-Part Systems: A Review of Problem Settings and Solution Approaches.

Schubert, D.; Scholz, A. & Wäscher, G. (2017): Integrated Order Picking and Vehicle Routing with Due Dates. Working Paper No. 7/2017, Faculty of Economics and Management, Otto-von-Guericke University Magdeburg.

Theys, C.; Bräysy, O.; Dullaert, W. & Raa, B. (2010): Using a TSP Heuristic for Routing Order Pickers in Warehouses. *European Journal of Operational Research* 200, 755-763.

Tompkins, J. A.; White, J. A.; Bozer, Y. A. & Tanchoco, J. M. A. (2010): Facilities Planning. 4th edition, John Wiley & Sons, New Jersey.

Tsai, C.-Y.; Liou, J. J. H. & Huang, T.-M. (2008): Using a Multiple-GA Method to Solve the Batch Picking Problem: Considering Travel Distance and Order Due Time. *International Journal of Production Research* 46, 6533-6555.

Wäscher, G. (2004): Order Picking: A Survey of Planning Problems and Methods. *Supply Chain Management and Reverse Logistics*, Dyckoff, H.; Lackes, R. & Reese, J. (eds.), 323-347, Springer: Berlin.

*A former version has been published in the FEMM working paper series.

Part II:

Literature Review

Picker Routing in Manual Picker-to-Part Systems: A Review of Problem Settings and Solution Approaches

A. Scholz, G. Wäscher

Abstract

In manual picker-to-part order picking systems, human operators (order pickers) walk or ride through the warehouse using a picking device in order to retrieve items which are specified by customer orders. The major part of the working time, an order picker spends for traveling through the warehouse. Therefore, finding short picker tours is pivotal for an efficient organization of warehouse operations. The construction of picker tours is part of the Picker Routing Problem (PRP). The PRP is characterized by the arrangement of the storage locations in the picking area (layout). In the first part of this paper, attributes regarding the layout are pointed out which affect the types of decisions to be made and the time complexity of the solution approaches to the respective PRPs. In the second part, the integration of the PRP into the Order Batching Problem (OBP), which deals with the grouping of customer orders into picking orders, is considered. Since the PRP and the OBP always arise simultaneously, an integrated solution of both problems has received much attention in the recent literature. However, solution approaches are rarely based on the same settings. Therefore, the algorithms are classified according to the underlying assumptions here in order to obtain a structured overview. Finally, for both PRPs and integrated problems, research gaps are identified.

Keywords: Order Picking, Picker Routing, Order Batching

Corresponding author:

André Scholz

Postbox 4120, 39016 Magdeburg, Germany

Phone: +49 391 67 51841

Fax: +49 391 67 48223

Email: andre.scholz@ovgu.de

1 Introduction

Order picking is a warehouse function responsible for the satisfaction of a given demand specified by customer orders. It deals with the retrieval of requested items from their storage locations in the picking area of the warehouse (Petersen & Schmenner, 1999; Wäscher, 2004). Order picking is necessary since articles are received and stored in large volumes, while customers request small volumes of different articles only. In manual order picking systems, which are prevalent in practice (de Koster et al., 2007), the picking process is performed by human operators (order pickers). Among manual picking systems, picker-to-part systems are the most important ones (de Koster, 2008). In those systems, order pickers walk or ride through the picking area using a picking device in order to retrieve requested items (Wäscher, 2004). Due to the employment of human operators on a large scale, order picking is considered to be the most cost-intensive warehouse function, as picking operations account for between 50% (Frazelle, 2002) and 65% (Coyle et al., 1996) of the total warehouse operating costs.

The picking process executed by the order pickers is mainly composed of traveling through the picking area, searching for the respective items and picking them from their storage locations. Traveling consumes 50% of an order picker's working time (Tompkins et al., 2010) and is the most important component. Therefore, retrieving requested items in such a way that the travel time is kept at a low level is pivotal for an efficient organization of warehouse operations. This gives rise to the so-called Picker Routing Problem (PRP). The PRP deals with the determination of a sequence according to which requested items are to be retrieved such that the distance to be covered by the order pickers is minimized. The PRP is characterized by the underlying layout of the warehouse, i.e. the arrangement of the storage locations in the picking area. Depending on the layout, the corresponding PRP can be solved efficiently or it is rather difficult to solve. In this paper, different criteria are identified for the classification of layouts first and existing solution approaches to the PRP are presented for each class of layouts. By doing so, the impact of the layout on the types of decisions to be made and on the complexity of the corresponding PRPs is pointed out.

More recently, it has been demonstrated that the picking process can further be improved by integrating the PRP into related planning problems. The Order Batching Problem (OBP) can be considered as the most popular problem predestined to be solved jointly with the PRP. The OBP deals with determining which customer orders are to be processed on the same tour and it always arises simultaneously with the PRP. Nevertheless, for a long time, the PRP did not receive much attention when dealing with the OBP. Only in recent years, first solution approaches have been developed which simultaneously tackle both problems. However, almost all approaches rely on different assumptions, which makes it very difficult to compare the performance of the algorithms. In order to give a comprehensive overview of the solution approaches, the main assumptions are pointed out and used for the classification of the algorithms to the respective problem variant. Furthermore, for each approach, results of numerical experiments are considered and the maximum size of the problem instances, as well as computing times required to solve the instances, are addressed.

The remainder of this paper is organized as follows: The next section comprises an overview of typical warehouse areas. The picking area and its characteristics are described in more detail before the order picking process is illustrated and planning problems arising in the picking process are mentioned. Section 3 is devoted to the PRP. Based on the type of the layout, different variants of the PRP are presented and corresponding solution approaches are explained. In Section 4, the integration of the PRP into the OBP is considered. Solution approaches are reviewed and classified based on their underlying assumptions. The paper concludes with an outlook on promising areas for future research (Section 5).

2 Manual picker-to-part order picking systems

2.1 Order picking warehouses

The basic processes in a warehouse involve (Gu et al., 2007) the receiving of shipments from suppliers, the storage of the respective items, the retrieval of stored items, and the preparation of retrieved items for shipment to the customers (see Fig. 1). Incoming shipments arrive by trucks at the receiving area, where the items are unloaded and either directly transferred to the shipping area or transported to the storage area of the warehouse. The storage area typically consists of two parts (Rouwenhorst et al., 2000), namely the reserve and the picking area. In the reserve area, huge amounts of items are stored in the most economical way until they are required for the replenishment of the inventory of the picking area. The picking area contains smaller volumes of items which are stored in such a way that they can easily be retrieved (picked). After the retrieval, the items are prepared for shipment and transferred to the shipping area from where they are transported to the respective customers.

Among all warehouse operations, the operations performed in the picking area are considered as the most cost-intensive ones (Gu et al., 2007). In the picking area, pallets, bins or low-level racks are typically used to store the items (de Koster et al., 2007). The arrangement of the storage locations determines the so-called layout of the picking area. In general, the picking area includes two types of aisles, namely picking aisles and cross aisles. Picking aisles have to be entered in order to retrieve requested items, as the storage locations are arranged on one side or even both sides of the picking aisles. Cross aisles do not contain any storage locations, but they are required in order to proceed from one picking aisle to another. If the picking aisles are straight, arranged parallel to each other, of identical length and width, and intersected by cross aisles at right angles, the layout is called conventional. In conventional layouts, the cross aisles divide the picking area into blocks and picking aisles into subaisles (see Fig. 2a)). If cross aisles only exist at the front and the rear of the picking area, the arrangement of the storage locations follows a single-block layout. Otherwise, at least one additional middle cross aisle exists, resulting in a multi-block layout.

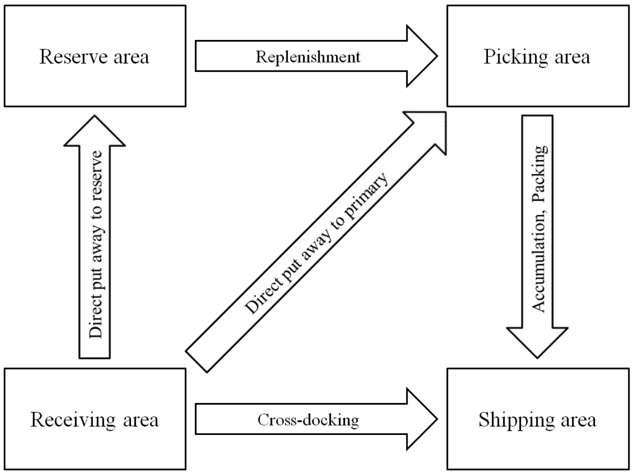


Fig. 1: Typical warehouse areas and flows (de Koster et al., 2007)

If the picking and cross aisles do not show the above-mentioned characteristics, the layout is called non-conventional. The most prominent non-conventional layouts are the flying-V and the fishbone layout (Gue & Meller, 2009). The flying-V layout is characterized by a curved cross aisle, where the angle between the cross aisle and an intersecting picking aisle gets larger the farther the picking aisle is away from the depot (see Fig. 2b)). A disadvantage of such a layout can be seen in the rather sharp turns that order pickers have to perform when entering the lower part of a picking aisle. This can be avoided by allowing picking aisles to be arranged both vertically and horizontally resulting in a fishbone layout (see Fig. 2 c)).

Besides the orientation of the picking aisles and the cross aisles, the width of the picking aisles represents an important characteristic of the picking area. Standard, wide and narrow picking aisles have to be distinguished. In standard aisles, items can be retrieved from both sides of the aisle without performing additional movements. At the same time, aisles are wide enough for order pickers to pass each other (see e.g. Roodbergen (2001)). In wide aisles, order pickers are also able to pass or overtake each other, whereas additional movements are required for picking items from storage locations of different sides of a picking aisle (Goetschalckx & Ratliff, 1988). When narrow aisles have to be dealt with, no such additional movements have to be executed. However, order pickers may interfere (block) each other as passing and overtaking is not possible in narrow aisles (Parikh & Meller, 2009).

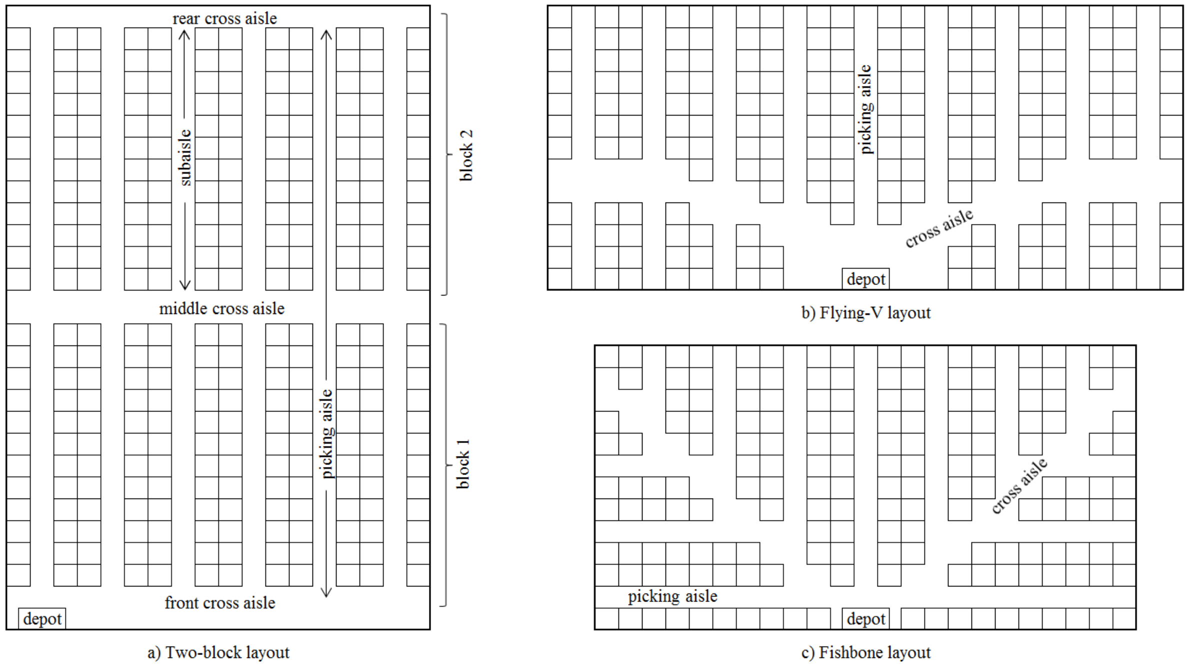


Fig. 2: Conventional and non-conventional layouts

The picking area either contains a single depot (centralized depositing) or retrieved items can be deposited at the front end of each picking aisle (decentralized depositing). Furthermore, it has to be distinguished between picking areas, where each article has exactly one storage location and warehouses, where multiple locations are assigned to certain articles.

2.2 Picking process

In manual picker-to-part systems, order pickers perform tours through the picking area of the warehouse in order to retrieve requested items from their storage locations. The information about which article is requested and how many items of this article are to be retrieved is comprised of a set of external or internal customer orders. Based on the customer orders, pick lists are generated which guide the order pickers through the warehouse. A pick list identifies the sequence according to which the storage locations of requested items (pick locations) are meant to be visited, and it contains information about the quantity to be picked at the respective pick locations. By means of a picking device (e.g. a cart or a roll cage), the order picker is able to temporarily store retrieved items, allowing the picker to retrieve several items on the same tour. Requested items retrieved on the same tour are addressed as a picking order (batch). The number of items which can be retrieved on the same tour is limited by the capacity of the picking device. The capacity may be specified in terms of a maximum number of items, a maximum weight or even a maximum number of customer orders.

At the end of the picking process, requested items have to be sorted according to their corresponding customer order as complete orders are allowed to be shipped to customers only. Two picking strategies can be distinguished: pick-and-sort and sort-while-pick (de Koster et al., 2007). Using the first strategy, items can be retrieved independently of the customer order to which they belong, resulting in an additional sorting effort after the items have been deposited. When applying the sort-while-pick strategy, all requested items of a customer order have to be picked on the same tour, i.e. splitting of customer orders is not permitted.

2.3 Planning issues

Due to the large proportion of time-consuming manual operations, minimization of the total time required for processing the orders, i.e. the time needed for performing the corresponding tours, is of prime importance and a common objective in order picking warehouses (de Koster et al., 2007). The time that an order picker spends for retrieving all items of a batch (batch processing time) can be divided into (Tompkins et al., 2010) the time for preparing a batch (setup time), the time required for traveling from the depot, to and between the pick locations and back to the depot (travel time), the time at the pick location for identifying the correct item (search time) and the time for the physical retrieval of the item (pick time). From these components, the travel time is of major importance as the other activities have to be performed anyway and are not dependent on the sequence according to which pick locations are visited within an order picker's tour (de Koster & van der Poort, 1998). Assuming the travel velocity of the pickers to be constant, the travel time is a linearly increasing function of the travel distance (Jarvis & McDowell, 1991). Consequently, minimizing the travel distance also minimizes the travel time.

Let a (non-empty) set of customer orders be given, each of which requiring certain items to be retrieved from the picking area of the warehouse. The distance to be covered for retrieving all requested items (total travel distance) is then determined by dealing with the following planning issues (de Koster et al., 2007):

- internal layout design (aisle configuration), i.e. the determination of the number of picking and cross aisles as well as their arrangement in the picking area (tactical level);

- storage assignment, i.e. the assignment of articles to storage locations (tactical and operational level);
- zoning, i.e. the assignment of aisles to work zones to which order pickers are restricted in their operations (tactical and operational level);
- order consolidation (order batching), i.e. the grouping of customer orders into batches (operational level);
- picker routing, i.e. the determination of a sequence according to which requested items of a batch are picked and the identification of the corresponding path through the picking area (operational level).

Obviously, the travel distance is affected by the sequence according to which pick locations are to be visited, i.e. the respective solution to the PRP. However, the decisions made regarding the other planning issues have a significant impact on the travel distance as well. Nevertheless, an integrated solution of all planning issues has not been considered in the literature because of two reasons. First, the resulting problem would be far too complex and second, the planning issues include decisions with different planning horizons (de Koster et al., 2007). The OBP is the only planning problem which always arises simultaneously with the PRP. Therefore, we focus on solution approaches to the PRP first and then proceed with the consideration of an integrated solution of the PRP and the OBP.

3 The Picker Routing Problem

The PRP can be formulated as follows (Ratliff & Rosenthal, 1983; Scholz et al., 2016): Given a set of items to be picked from known storage locations, in which sequence should the locations be visited such that the total length of the corresponding tour is minimized? The PRP can thus be interpreted as a special case of the Traveling Salesman Problem (TSP), while the special characteristic of the PRP can be found in the layout of the picking area, i.e. the width of the picking aisles (standard, wide, narrow) and the arrangement of the picking and cross aisles (single-block, multi-block, non-conventional layout).

In Table 1, an overview of solution approaches to the PRP is given. The first column includes the authors of the respective publication. The next three columns specify the characteristics of the picking area, where the columns give information about the width of the picking aisles, the arrangement of aisles and additional specifications, respectively. The fifth column contains a brief description of the solution approach. Whether an approach always provides an optimal solution (exact) or not (heuristic) can be seen in the sixth column. Furthermore, the seventh column shows the computational effort for algorithms as well as the number of variables and constraints (size) for mathematical programming

Table 1: Solution Approaches to the Picker Routing Problem

Reference	Aisle width	Layout	Additional characteristics	Solution approach	Type of approach	Complexity
Ratliff & Rosenthal (1983)	standard	single-block		Five possible paths for retrieving items in an aisle are pointed out. Based on dynamic programming, one path is chosen for each aisle.	exact	$O(m+n)$
Scholz et al. (2016)	standard	single-block		A problem-specific model formulation is given based on the algorithm of Ratliff & Rosenthal (1983).	exact	model size linear in m
Petersen (1997)	standard	single-block		Simple routing policies are applied: S-shape, return, midpoint, largest gap and composite heuristics.	heuristic	$O(m+n)$ for largest gap; $O(m)$ for other heuristics
de Koster & van der Poort (1998)	standard	single-block	decentralized depositing	The algorithm of Ratliff & Rosenthal (1983) is extended to the case of multiple deposit locations.	exact	$O(m+n)$
Daniels et al. (1998)	standard	single-block	multiple article locations	A tabu search algorithm is proposed.	heuristic	
Roodbergen & de Koster (2001a)	standard	two-block		The algorithm of Ratliff & Rosenthal (1983) is extended.	exact	$O(m+n)$
Scholz (2016)	standard	multi-block		The formulation of Scholz et al. (2016) is extended to the case of multiple blocks.	exact	model size linear in $q \cdot m$
Vaughan & Petersen (1999)	standard	multi-block		Aisle-by-aisle heuristic: Based on dynamic programming, the cross aisles are determined used to enter or leave a picking aisle.	heuristic	$O(q^2 \cdot m)$
Roodbergen & de Koster (2001b)	standard	multi-block		The routing strategies S-shape, return and largest gap are extended to the case of multiple blocks.	heuristic	$O(q \cdot m)$ for S-shape or return; $O(q \cdot (m+n))$ for largest gap
Roodbergen & de Koster (2001b)	standard	multi-block		Combined heuristic: Based on dynamic programming, the cross aisles are determined used to enter or leave a subaisle.	heuristic	$O(q \cdot m)$
Theys et al. (2010)	standard	multi-block		Different TSP heuristics are applied to the PRP.	heuristic	polynomial in n
Çelik & Süral (2014)	standard	fishbone & flying-V		The algorithm of Roodbergen & de Koster (2001a) and routing strategies for PRPs with three blocks are modified.	exact & heuristic	same as the respective basic routing algorithms
Goetschalckx & Ratliff (1988)	wide	single-block		Four strategies for retrieving items in a wide aisle are integrated into the algorithm of Ratliff & Rosenthal (1983).	exact	$O(m+n^2)$
Chen et al. (2013)	narrow	multi-block		An ant colony approach is proposed for the case of two pickers.	heuristic	
Chen et al. (2016)	narrow	multi-block		An ant colony approach is proposed for the case of an arbitrary number of pickers.	heuristic	

formulations. Both the computational effort and the size of a model may be dependent on the number of blocks q, the number of picking aisles m and the number of pick locations n. Note that no information is given for metaheuristic approaches as they are terminated after a fixed time limit or after the execution of a certain number of iterations. The approaches depicted in Table 1 are explained in greater detail in the following subsections, starting with solution approaches to PRPs in standard-aisle warehouses.

3.1 The Picker Routing Problem in standard-aisle warehouses

Single-block layout

The single-block layout represents the simplest form of conventional layouts and has frequently been assumed in the literature so far. It is characterized by the existence of exactly two cross aisles, one at the front and one at the rear of the picking area. Thus, an order picker has only two possibilities for switching between picking aisles. Using the special structure of the picking area, Ratliff & Rosenthal (1983) developed an efficient algorithm able to optimally solve any practical-sized PRP in a single-block layout within fractions of a second. They pointed out that picking aisles can be considered successively and proved that only five options have to be taken into account for the retrieval of items from the storage locations of a picking aisle. A graph representing the tour is constructed. Starting with a graph with an empty set of edges, edges are added corresponding to the picking aisles from left to right. By means of dynamic programming, one out of the five options is chosen for each picking aisle, resulting in an optimal solution. In this approach, a constant number of graphs has to be considered for each picking aisle. Thus, the computational effort of the algorithm increases only linearly with the number of picking aisles. Due to the construction process regarding the five options, the increase of the computational effort is also linear in the number of pick locations.

Another exact approach has been proposed by Scholz et al. (2016), who developed a problem-specific mathematical model to the PRP. First, a graph to the PRP is constructed based on an observation of Burkard et al. (1998) who formulated the PRP as a Steiner TSP. In this representation, the set of Steiner points, i.e. the vertices which do not have to be visited, contains the locations of the intersections between a picking and a cross aisle. The remaining vertices are given by the location of the depot and the pick locations. Taking the structure of optimal solutions to the PRP into account, the graph is modified in such a way that its size (in terms of the number of vertices and edges) is totally independent of the number of pick locations. A TSP formulation is then applied to this graph, resulting in a model formulation whose size linearly increases with the number of picking aisles. By application of a commercial IP-solver, any practical-sized PRP instance can be solved within a small amount of computing time (Scholz et al., 2016). Furthermore, it is shown that application of this model outperforms the usage of general TSP and Steiner TSP formulations by far in terms of computing times and optimal solutions obtained within a given time limit.

Although very fast exact approaches exist, the application of simple heuristic routing strategies is prevalent in practice (Roodbergen, 2001). This can be explained by the fact that tours resulting from such routing strategies are more straightforward and can be memorized easily, whereas optimal tours seem to be quite confusing for order pickers, increasing the risk of missing a requested item (Petersen & Schmenner, 1999). The S-shape, return, midpoint and largest gap strategies represent such simple routing policies. Following the S-shape or the return strategy, picking aisles are visited from left to right. As for the S-shape strategy, each picking aisle containing at least one requested item is traversed. An exception may occur in the last picking aisle to be visited. If this aisle is entered from the front cross aisle, the order picker moves to the pick location farthest away from the front cross aisle and then returns for retrieving the remaining requested items in that aisle. According to the return strategy, each picking aisle is entered and left via the front cross aisle in such a way that all requested items are collected. When applying the midpoint or the largest gap strategy, each picking aisle is divided into a lower and an upper part. The order picker traverses the leftmost picking aisle containing a pick location and then visits the picking aisles, from which an item has to be picked, from left to right, retrieving all requested items located in the upper part of the picking area. The rear cross aisle is used for entering and leaving the respective picking aisles. When reaching the rightmost picking aisle with pick locations, the order picker traverses this aisle in order to reach the front cross aisle from where the remaining requested items are retrieved. The midpoint and the largest gap policy only differ in the way how the picking area is divided into the two parts. As for the midpoint strategy, the distance of a pick location to the front cross aisle is considered. If the distance is shorter than half of the length of the picking aisle, the pick location is assigned to the lower part of the warehouse. Otherwise, it belongs to the upper part. When applying the largest gap strategy, the largest distance (gap) between two adjacent pick locations or a pick location and the adjacent cross aisle is determined for each picking aisle. Pick locations from below the largest gap are assigned to the lower part, while the upper part includes the remaining requested items. As can be seen, both the midpoint and the largest gap strategy result in tours in which picking aisles may be visited twice. However, since items in a picking aisle are retrieved in such a way that the non-traversed distance is maximal when applying the largest gap policy, this strategy outperforms the midpoint policy in terms of solution quality (Hall, 1993). In Fig. 3, an example for an optimal tour as well as tours obtained by using the S-shape, return and largest gap strategies are depicted. (Note that the tour constructed according to the midpoint policy matches with the tour shown in Fig. 3d).) The storage locations of requested items are symbolized by black rectangles.

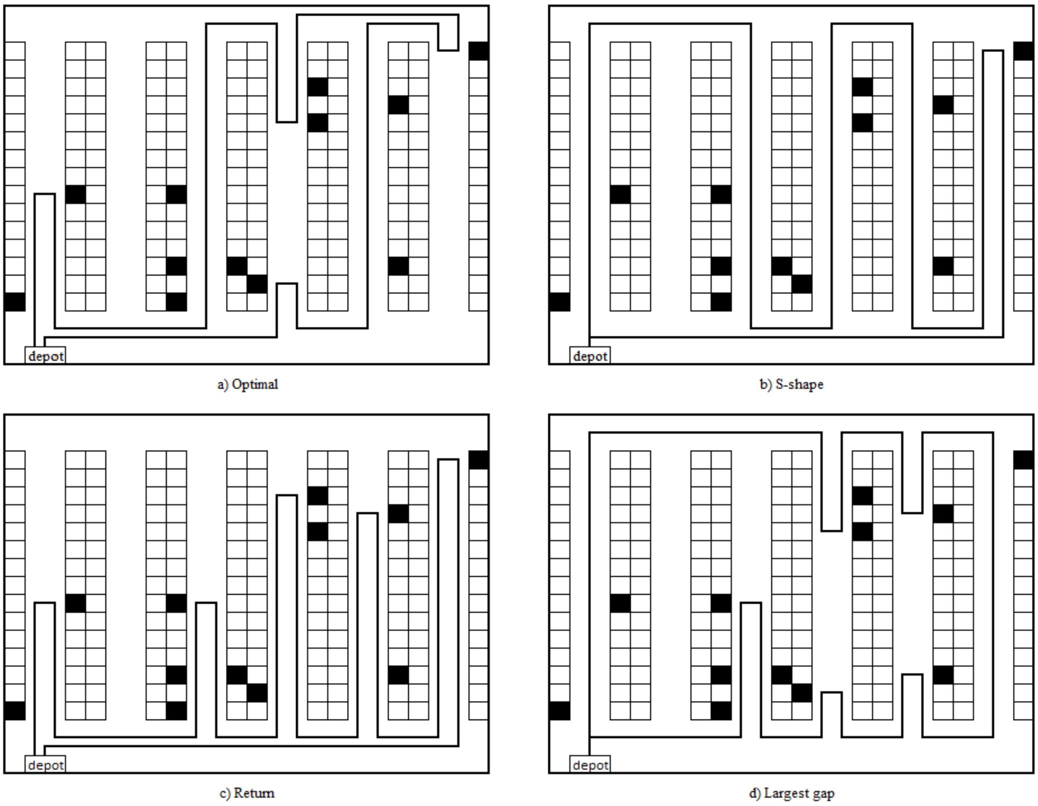


Fig. 3: Example picker tours in a single-block layout

As can be seen in Fig. 3, in comparison to an optimal tour, tours generated by means of the routing strategies may appear to be very simple. However, the solution quality of these routing policies is strongly dependent on the problem data (e.g. the number of picking aisles and pick locations) and in many situations, application of simple routing strategies results in tours with tour lengths far from the optimum (Roodbergen, 2001). Therefore, other routing strategies have been designed which tend to result in shorter tours while still having a simple structure. In this context, simple means that each picking aisle is visited at most once, i.e. when retrieving the items in a picking aisle, the picker either traverses the aisle or returns at the pick location farthest from the cross aisle from where the picking aisle has been entered. Thus, the resulting routing strategies combine elements of the S-shape and the return policy. Petersen (1997) was the first who proposed such a routing strategy called composite heuristic. Following this strategy, for each picking aisle, it is independently determined whether the distance to be covered is smaller for the application of a return move or for a move according to the S-shape strategy. The shorter one is executed. A more sophisticated routing heuristic has been developed by

Vaughan & Petersen (1999). In the so-called aisle-by-aisle heuristic, the type of move to be executed in a picking aisle is determined by means of dynamic programming. This approach tends to result in shorter tours than the application of the composite heuristic. However, although such tours still have a simple structure, they may not appear straightforward for order pickers as they have been generated by means of dynamic programming and do not follow a simple rule.

In all approaches mentioned above, it is assumed that the pickers have to return to the depot in order to unload retrieved items. However, in modern warehouses with paperless work, there is no need for order pickers to return to the depot each time the items of a batch have been picked. Instead, requested items could be deposited at the head of an arbitrary picking aisle and new instructions could be received by means of a mobile computer. This allows the order pickers to process several batches in a row resulting in significant time savings. This variant of the PRP has been considered by de Koster & van der Poort (1998). They modified the algorithm of Ratliff & Rosenthal (1983) by adding a vertex for each picking aisle representing the deposit location at the head of the aisle. The general concept of the algorithm remains unchanged and the computational effort is still linear in m and n.

Instead of multiple deposit locations, Daniels et al. (1998) dealt with PRPs in picking areas where articles can be assigned to multiple storage locations, respectively. When an item of an article has to be picked, it has then to be decided which storage location is included in the tour. Furthermore, Daniels et al. (1998) introduced an inventory level at each storage location, i.e. multiple locations may have to be visited for retrieving all items of a certain type. The authors formulate the problem as a modified TSP. For solving the problem, three TSP heuristics, namely the nearest-neighbor and the shortest-arc heuristic as well as a randomized construction approach are presented. These approaches are modified in such a way that inventory levels and quantities picked are taken into account. For the generation of high-quality solutions, Daniels et al. (1998) designed a tabu search approach where moves according to the neighborhood structures exchange a certain number of locations included in the tour.

Multi-block layout

Using a single-block layout is rarely the best choice for designing a picking area, as the introduction of additional middle cross aisles enables order pickers to switch between picking aisles at several positions, resulting in much shorter tours (Roodbergen et al., 2008). However, when dealing with multiple blocks, the determination of optimal tours, as well as the structure of tours in general, gets much more complex.

For solving the PRP in a two-block layout, Roodbergen & de Koster (2001a) managed to extend the algorithm of Ratliff & Rosenthal (1983). The computational effort of the algorithm still linearly increases with the number of picking aisles and pick locations. However, in the approach of Roodbergen & de Koster (2001a), the number of graphs to be constructed in each iteration is much larger. While the number of graphs amounts to 50 in the approach of Ratliff & Rosenthal (1983), up to 331 graphs have to be constructed in an iteration of the algorithm of Roodbergen & de Koster (2001a). Moreover, it would be very difficult to further extend the algorithm to PRPs including more than two blocks (Roodbergen, 2001). Up to now, no efficient algorithm exists which can deal with PRPs in picking areas with a three-block or even an arbitrary multi-block layout. Therefore, Scholz (2016) extended the formulation of Scholz et al. (2016) to the case of multiple blocks. In order to keep the size of the model at a reasonable level, several procedures are applied which significantly reduce the size of the underlying graph. The model formulation is suitable for solving PRPs with an arbitrary number of blocks as computing times do not increase if more blocks are considered (Scholz, 2016).

As it is the case for the single-block layout, heuristic approaches are frequently used to deal with PRPs in multi-block layouts. The aisle-by-aisle heuristic can also be applied to multi-block layouts (Vaughan & Petersen, 1999). Each picking aisle is visited once and by means of dynamic programming, the cross aisles used for entering or leaving the aisle are determined, respectively. Furthermore, Roodbergen & de Koster (2001b) extended the routing strategies previously presented to the case of multiple blocks. In the extended version, the blocks are successively considered, starting from the block farthest from the depot. The respective routing strategy is then applied to the block under consideration before proceeding with the next block. Thus, the general concept of the respective routing strategy remains unchanged. However, with an increasing number of blocks, tours become much more complex and the solution quality further deteriorates (Roodbergen, 2001). The problem-specific heuristic approach, which leads to the best solutions in most settings has been proposed by Roodbergen & de Koster (2001b) and is called combined heuristic. This heuristic is similar to the aisle-by-aisle heuristic as dynamic programming is applied in order to determine which cross aisles are to be used. A difference can be seen in the fact that subaisles instead of complete picking aisles are considered in this approach. The order picker starts from the depot and traverses the leftmost picking aisle to be visited up to the block farthest from the depot. The aisle-by-aisle heuristic is then applied to this block. After having retrieved all requested items in the block, the picker goes to the next block again following the aisle-by-aisle heuristic. In comparison to the aisle-by-aisle heuristic, the combined heuristic is particularly advantageous when picking aisles are long, and it provides good solutions even for a larger number of blocks (Roodbergen, 2001). In order to further improve the solution quality, Roodbergen & de Koster (2001b) modified the combined heuristic with respect to the movements in the block nearest to the depot (block 1). In its original version, the leftmost picking aisle containing requested items is used to go to the block farthest from the depot. Now, the picker is permitted to deviate from this path and retrieve all items from block 1 located in the left of picking aisle $\tilde{m}$ before proceeding to the block farthest from the depot. Optimizing over $\tilde{m} \in \{1,\ldots,m\}$ generates a tour not longer than the original tour constructed by the combined heuristic.

Apart from problem-specific heuristic approaches, TSP heuristics have been applied to the PRP in a multi-block layout by Theys et al. (2010). The authors pointed out that the Lin-Kernighan-Helsgaun (LKH) heuristic (Helsgaun, 2000) provides solutions of outstanding quality. It reduces the tour length obtained by application of the S-shape strategy by up to 48%. With respect to the solution quality, the LKH heuristic represents the best heuristic which has been applied to the PRP.

Non-conventional layouts

More recently, other designs than conventional layouts have been considered and situations have been identified in which using such non-conventional layouts is advantageous. Çelik & Süral (2014) proposed an exact approach to the PRP in flying-V and fishbone layouts. First, the authors represent the PRP as a Steiner TSP as previously described. The graph is then transformed in such a way that its structure corresponds to the Steiner TSP representation of a PRP in a two-block layout. The exact algorithm of Roodbergen & de Koster (2001a) is applied to construct an optimal tour in a fishbone or flying-V layout. Çelik & Süral (2014) also adapted the S-shape, largest gap and aisle-by-aisle strategies to the PRP in fishbone layouts. The picking area is divided into three regions: the horizontal picking aisles left from the depot, the vertical picking aisles and the horizontal picking aisles located on the right of the depot. These different regions are then treated as different blocks and the routing strategies are applied as to a PRP in a three-block layout. Çelik & Süral (2014) compared fishbone and conventional layouts with respect to the distance to be covered for retrieving a set of items. For both layout types, they computed optimal solutions for different settings and pointed out that distance savings by up to 20% can be achieved using fishbone layouts. However, this observation holds for pick lists including only one or two items which was expected as the average distance between storage locations and the depot is smaller in fishbone layouts. With an increasing size of the pick list, the advantage of fishbone layouts diminishes. For pick lists with 30 items, tours are up to 36% longer than in conventional layouts.

3.2 The Picker Routing Problem in wide-aisle warehouses

When dealing with standard aisles, it is assumed that the order picker can retrieve items from both sides of the picking aisles without consuming additional time. In practice, the picker often cannot reach both sides without changing the position as picking aisles are four meters wide or even more (Goetschalckx & Ratliff, 1988). In standard-aisle warehouses, moves related to the S-shape, the return or the largest gap strategies enable order pickers to retrieve all items in a picking aisle. When using return or largest gap moves in a wide picking aisle, all items can be picked as well. The order picker starts with picking all items from one side and then returns while retrieving the requested items from the other side. Regarding S-shape moves, two possibilities have to be considered in wide-aisle warehouses. Either the picking aisles is traversed twice (split traversal strategy) collecting the items from one side, respectively, or it is traversed once (traversal strategy) in such a way that all items are retrieved. Based on the four possible movements which can be performed in a picking aisle, Goetschalckx & Ratliff (1988) modified the algorithm of Ratliff & Rosenthal (1983) to PRPs in wide-aisle warehouses. In order to determine an optimal tour, for each picking aisle, the minimum distance to be covered for application of the traversal strategy has to be calculated. Goetschalckx & Ratliff (1988) reduced the problem to finding a shortest path in an acyclic graph by means of dynamic programming. This is done in $O\left(\tilde{n}^2\right)$ time, where $\tilde{n}$ denotes the number of pick locations in the respective picking aisle. The computational effort of the modified algorithm of Ratliff & Rosenthal (1983) then amounts to $O\left(m+n^2\right)$, where m and n denote the number of picking aisles and pick locations, respectively. Thus, it can be observed that PRPs in wide-aisle warehouses seem to be more difficult to solve than PRPs in warehouses with standard aisles. However, only minor modifications are required for adapting approaches to the PRP with standard aisles to the case of wide aisles.

3.3 The Picker Routing Problem in narrow-aisle warehouses

Due to limited space in the picking area, very narrow picking aisles have to be dealt with in many practical applications (Gu et al., 2007). In case of narrow picking aisles, order pickers can neither pass nor overtake each other, i.e. they may have to wait until their path is not blocked by another order picker. This results in three main differences compared to the PRPs in standard-aisle warehouses. First, tours of different pickers cannot be constructed independently of each other. Second, it is not sufficient to determine the path through the warehouse but waiting instructions may have to be given to the order pickers. Waiting instructions include information about which picker has to wait at which point in time for how long. Third, the minimization of the total travel distance does not represent a valid objective in narrow-aisle warehouses since short tours do not guarantee for short processing times. Thus, it can be concluded that the PRP in warehouses with narrow aisles significantly differs from the standard-aisle case. Due to the interdependencies of the tours of different pickers, PRPs in narrow-aisle warehouses are much more difficult to solve and no efficient solution approach exists so far.

Chen et al. (2013) were the first who designed a metaheuristic approach to the PRP in narrow-aisle warehouses. They considered a scenario where a given set of customer orders is processed by two order pickers. The sequence according to which the orders are processed is given. The objective is to minimize the average throughput time of an order which is defined as the difference between the completion date of an order and its arrival date, i.e. the point in time when the order has become available at the warehouse. In order to solve this problem, Chen et al. (2013) proposed an ant colony optimization (ACO) approach. The tour corresponding to the first order to be processed is assigned to the first picker and is constructed without consideration of blocking. This tour will remain unchanged. The tour of the second picker is then determined while taking the tour of the first picker into account, i.e. waiting instructions may be given to the second picker. If more than two orders exist, the next two orders are not processed before both pickers have returned to the depot. In the numerical experiments, Chen et al. (2013) applied the ACO approach to instances with two orders comprising up to 30 pick locations. Solving an instance of this size required 10 seconds of computing time. However, the solution quality of the algorithm was barely superior to the quality of solutions obtained by application of a modified S-shape strategy.

Chen et al. (2016) extended the approach of Chen et al. (2013) to the case of an arbitrary number of order pickers. As in Chen et al. (2013), it is assumed that each order picker processes one order, then returns to the depot and waits until all pickers have finished their work. First, by means of an ACO approach, a tour is constructed for each picker without taking blocking aspects into account. In a second step, instructions are given to order pickers if blocking situations arise. If pickers block each other by picking items in the same aisle, then the picker who first enters the picking aisle performs the tasks and waiting instructions are given to the other pickers. If a blocking situation is caused by a picker traversing an aisle without retrieving items, the order picker can be instructed to use another aisle. Chen et al. (2016) applied the algorithm to instances with up to 10 order pickers and 30 pick locations per order. Unfortunately, computing times have not been reported. As it is the case for the basic algorithm proposed by Chen et al. (2013), the approach of Chen et al. (2016) is not able to significantly improve solutions provided by modified S-shape and largest gap strategies.

4 Order Batching and Picker Routing

Order Batching and Picker Routing Problems both represent planning problems at the operational level. They always arise simultaneously in practical applications. Nevertheless, these problems have been treated separately for a long time. In fact, the PRP has even been neglected completely, i.e. very simple routing policies have been applied only, and all effort has been put in solving the OBP. More recently, the benefit of solving the PRP and the OBP simultaneously has been identified and a large variety of solution approaches to the Joint Order Batching and Picker Routing Problem (JOBPRP) have been proposed. The JOBPRP can be defined as follows (Scholz & Wäscher, 2017): Let a set of customer orders be given, each of which including certain items to be retrieved from known storage locations. A picking device with limited capacity is used for collecting requested items. The following two questions have then to be dealt with.

- How should the set of customer orders be grouped into batches? (order batching)
- For each batch, in which sequence should the items included be retrieved? (picker routing)

How difficult the JOBPRP is to solve mainly depends on the objective. Objectives to the JOBPRP can be divided into distance-related and tardiness-related objectives. In the first case, the tours are constructed in such a way that the length of all tours (total tour length) is minimized. In the letter case, a due date is assigned to each customer order and these due dates are to be met in the best possible way. A very common tardiness-related objective represents the minimization of the total tardiness, i.e. the extent to which the due dates are violated (Henn & Schmid, 2013; Chen et al., 2015). Solving the JOBPRP regarding a tardiness-related objective is much more complex because it is not sufficient to group orders into batches but batches have also to be assigned to order pickers and for each order picker, a sequence has to be determined according to which the batches assigned to the picker are to be processed. Thus, the number of order pickers is also an important date, which is not the case when dealing with distance-related objectives.

Independent of the objective to be dealt with, solution approaches to the JOBPRP typically have the same structure consisting of two components. The first component is a metaheuristic regarding the batching problem. In this component, the composition (and assignment and sequence) of batches is modified in order to obtain a better solution. The second component contains the routing algorithm and is only used for the evaluation of solutions. The two components, as well as information about the problem settings and the maximum size of the instances (in terms of the number of customer orders, the number of requested items per order and the capacity of the picking device) considered in the numerical experiments, are depicted in Table 2 for each solution approach.

4.1 Distance-related objectives

Most approaches to the JOBPRP deal with the minimization of the total travel distance. However, almost all approaches rely on different assumptions regarding the problem settings, i.e. the measurement of the capacity, if splitting of customer orders allowed or not and the layout of the picking area, which makes it impossible to compare the performance of the algorithms. In the following, the solution approaches are reviewed based on how the capacity of the picking device is determined.

Maximum number of orders

If orders consist of a relatively low or an almost identical number of items, order pickers usually use picking devices with bins for performing their tours. Items belonging to the same customer order are then placed in the same bin (Gademann & van de Velde, 2005), implying that the maximum number of orders processed on the same tour equals the number of bins.

Cheng et al. (2015) proposed a particle swarm optimization (PSO) approach to the JOBPRP with a capacity limited by the number of orders. In PSO approaches, a population of solutions (particles) is encoded and moved with a certain velocity around the search space, guided by its own position and the position of the particle representing the best known solution. The authors used an encoding scheme which can be divided into two parts. The first part gives information about the number of orders contained in the batches, while the second part arranges the orders into a sequence. As the size of each batch is given by the first part, the sequence also determines the composition of the batches. For the generation of an initial population, the authors apply a random procedure to establish the batch sizes. Based on the proximity of the storage locations in the order, the order sequence is generated. The objective function value of a solution is determined by representing the arising routing subproblem as a TSP and applying an ACO approach. By means of numerical experiments, Cheng et al. (2015) showed that this approach provides optimal or near-optimal solutions within a few seconds of computing time for small instances with up to 7 customer orders. For solving large instances with 200 orders, computing times of up to 20 minutes are required.

Lin et al. (2016) dealt with the same problem and also proposed a PSO approach. For the encoding of a solution, the warehouse is represented as a grid consisting of storage locations and locations in picking and cross aisles. Each order is then represented by a single location (order center) in the grid. The order center denotes the location with the smallest distance to all pick locations included in the respective order. A batch center is analogously defined. Thus, coordinates of order centers are known, whereas

Table 2: Solution Approaches to the Joint Order Batching and Picker Routing Problem

Citation	Objective(s)	Problem settings	Batching approach	Routing approach	Instance size
Cheng et al. (2015)	minimization of total tour length	capacity: number of orders; layout: arbitrary	particle swarm optimization	ant colony optimization	up to 200 orders; 4 to 10 items per order; up to 5 orders per tour
Lin et al. (2016)	minimization of total tour length	capacity: number of orders; layout: arbitrary	particle swarm optimization	nearest-neighbor heuristic	100 orders; 1 to 16 items per order; 4 orders per tour
Matusiak et al. (2014)	minimization of total tour length	capacity: number of orders; layout: arbitrary with several drop-off locations; precedence constraints for all items in an order	simulated annealing	exact A*-algorithm of Psaraftis (1980)	up to 150 orders; up to 50 items per order; 4 orders per tour
Won & Olafsson (2005)	minimization of total tour length and throughput time	capacity: number of items; layout: arbitrary; orders arrive with a certain rate	constructive approach with multiple starts	2-opt heuristic	10 orders per hour; 1 to 5 items per order; up to 10 items per tour
Scholz & Wäscher (2017)	minimization of total tour length	capacity: number of items; layout: two-block	iterated local search	algorithm of Roodbergen & de Koster (2001a) and routing strategies	up to 80 orders; 5 to 25 items per order; up to 75 items per tour
Kulak et al. (2012)	minimization of total tour length	capacity: weight of items; layout: arbitrary	tabu search	several TSP algorithms	up to 250 orders; 2 items per order on average; up to 50 items per tour
Grosse et al. (2014)	minimization of total tour length	capacity: weight of items; layout: single-block; splitting of orders allowed	simulated annealing	savings heuristic and return, midpoint and largest gap strategies	26 orders; 10 to 60 items per order; 50 items per tour on average
Tsai et al. (2008)	minimization of travel costs and earliness and tardiness penalties	capacity: weight of items; layout: single-block; splitting of orders allowed	genetic algorithm	genetic algorithm	up to 250 orders; 50 items per order on average; 3000 items per tour on average
Chen et al. (2015)	minimization of the total tardiness	capacity: number of orders; layout: arbitrary	genetic algorithm	ant colony approach	up to 8 orders; up to 10 items per order; up to 4 orders per tour
Scholz et al. (2017)	minimization of the total tardiness	capacity: number of items; layout: multi-block	variable neighborhood descent approach	combined and LKH heuristics	up to 200 orders; 5 to 25 items per order; up to 75 items per tour

coordinates have to be determined for batch centers. The coding scheme of the solution then consists of two parts. The first part contains a permutation of customer orders and the second one comprises the coordinates of each batch center. For decoding a solution, customer orders are successively considered and assigned to the batch with a positive remaining capacity whose batch center has the smallest distance to the order center. The arising PRPs are solved by means of the nearest-neighbor heuristic. Lin et al. (2016) conducted numerical experiments for the evaluation of the impact of different algorithmic parameters only. No comparison to other approaches is given. Dependent on the number of particles used in the PSO approach, the computing time for solving an instance with 100 customer orders varies between 20 seconds and 6 minutes. The approach does not seem to be as time-consuming as the PSO approach by Cheng et al. (2015). However, application of the simple nearest-neighbor heuristic to the arising PRPs can be expected to have a significant negative impact on the solution quality.

Matusiak et al. (2014) integrated additional precedence constraints for the picking of a customer order. In this setup, items of an order have to be retrieved according to a predefined sequence. When all items of an order have been picked, the items have to be deposited at the drop-off location of the respective order. The picker returns to the depot when all orders in the batch have been processed. Matusiak et al. (2014) developed a simulated annealing (SA) approach to this variant of the JOBPRP. An initial solution is constructed by application of the savings heuristic (Clarke & Wright, 1964). Neighbor solutions are generated by means of the so-called REMIX procedure which randomly selects a certain number of batches and reassigns the orders contained in these batches. For dealing with the routing problems, Matusiak et al. (2014) used an A^*-algorithm similar to the approach of Psaraftis (1980) designed for a variant of the Dial-a-Ride Problem. The state of the algorithm is defined by a vector whose components indicate the last item picked for each order. The A^*-algorithm optimally solves the PRP with precedence constraints. However, the computational effort exponentially increases with the number of orders in a batch. An estimation method is applied when batches are composed of more than two orders. Nevertheless, solving an instance with 150 customer orders and a capacity of 4 orders per tour requires 3 hours of computing time.

Maximum number of items

When orders are highly heterogeneous with respect to the size, the picking device cannot be divided into equally-sized bins. Instead, the picking device only includes a single loading area where all items retrieved on the tour are stored. In this case, the capacity cannot be expressed in terms of the number of orders but rather is dependent on the loading space of the picking device and the capacity requirements of the items. If capacity requirements are fairly even for all items, the capacity can be expressed by a maximum number of items allowed to be included in a batch.

Won & Olafsson (2005) were the first ones who dealt with the JOBPRP considering this type of capacity constraint. They did not assume all customer orders to be known in advance. Instead, orders arrive at a certain rate. Besides the total travel time, the throughput time of all orders is minimized. Won & Olafsson (2005) proposed a constructive approach with multiple starts to the batching problem. First, minimum and maximum between-batch times ($t_{\min}$ and $t_{\max}$) are chosen. The between-batch time denotes the difference between the point in time an order is dispatched and its arrival date. A set of batches is constructed starting with all orders whose between-batch time is not larger than $t_{\min}$. The between-batch time is then incremented, resulting in another set of batches. This procedure is repeated until $t_{\max}$ is reached. The set of batches leading to the smallest objective function value is taken as the solution. The objective function value is determined by means of the 2-opt heuristic. Problem instances with up to 100 orders arriving per hour have been solved in the numerical experiments. Computing times have not been reported as they are negligible.

Scholz & Wäscher (2017) designed an approach to the JOBPRP aiming at the minimization of the total tour length. For the batching subproblem, an iterated local search (ILS) approach suggested by Henn et al. (2010) is adapted. An initial solution is constructed by means of the first-come-first-served heuristic. The improvement phase consists of two different neighborhood structures. In the first structure, a neighbor solution is generated by moving an order from one batch to another (shift), while orders between two different batches are exchanged (swap) in the second neighborhood. The perturbation phase interchanges a random number of customer orders between two batches. For the determination of the total tour length, different routing strategies as well as the exact algorithm of Roodbergen & de Koster (2001a) have been integrated, which makes the approach being restricted to a two-block layout. Scholz & Wäscher (2017) conducted numerical experiments to analyze if rather simple or more complex routing algorithms should be integrated into the batching heuristic when large instances are to be solved within a small amount of computing. Instances with up to 80 customer orders have been solved within 4 minutes of computing time. It is shown that exact routing outperforms heuristic strategies although far fewer iterations are performed in the batching algorithm.

Maximum total weight of items

As items stored in a warehouse are typically heterogeneous regarding their size and shape, the number of items is often not an appropriate measure for the capacity of the picking device (Grosse et al., 2014).

In order to provide a more realistic measure, a maximum total weight of items included in a batch is taken as the capacity. For example, it can be represented by the maximum weight until which the order picker is able to push the picking device without risking musculoskeletal disorders.

This kind of capacity constraints has been considered by Kulak et al. (2012), who proposed a tabu search (TS) algorithm to the batching problem. The construction of initial batches is based on so-called similarity indices. The similarity index of two batches i and j is defined as the ratio between the distance to be covered for retrieving all items of batch i and the distance covered for visiting all pick locations included in batches i and j, while tours are constructed by means of the nearest-neighbor heuristic. In the TS algorithm, neighbor solutions are generated by application of the same shift and swap moves as in the ILS approach of Henn et al. (2010). The arising routing problems are solved by means of the nearest-neighbor and Or-opt or the savings and 2-opt heuristics. This approach has proven to be very fast, generating solutions to instances with 250 customer orders in less than 2 minutes.

Grosse et al. (2014) made two additional assumptions regarding the problem settings. First, they allow orders to be split when being batched, i.e. items included in the same order may be assigned to different batches. Second, a single-block layout is assumed. Grosse et al. (2014) used the standard objective function and aimed for minimizing the total tour length. They suggested a SA algorithm and generated an initial solution by clustering items into batches based on different routing strategies. According to the neighborhood structure used in the SA approach, an item included in a batch is moved to another batch. The objective function value of a solution is determined by applying the same routing strategies which have been used for the initial clustering. Instances with an order size of up to 60 items have been solved within 20 minutes of computing time in the numerical experiments.

4.2 Tardiness-related objectives

Tsai et al. (2008) considered the same problem settings as Grosse et al. (2014) but they additionally introduced due dates for the orders and minimized the total costs arising from traveling and from completing orders too early or too late. Thus, a combination of a distance-related and a tardiness-related objective is considered. Assuming that one picker is available, they proposed a genetic algorithm for the batching problem in which a chromosome is divided into several gene segments. Each segment includes items of the same article and an allele represents the number of the batch to which the corresponding item is assigned. Tsai et al. (2008) used two-point crossover operations in which two gene segments of the parent chromosomes are randomly chosen and exchanged. By application of the mutation operation, the alleles of two randomly selected genes are exchanged. The fitness value of a chromosome is determined by application of another genetic algorithm solving the routing problems. In this algorithm, a chromosome gives the routing sequence, i.e. each allele represents the number of a pick location. A partially-matched crossover is applied, which means that two gene segments are exchanged between the parent chromosomes and the alleles are then modified in such a way that chromosomes representing feasible tours are generated. In the numerical experiments, instances with up to 250 orders have been solved. However, the solution approach consumes more than 3 hours of computing time for solving those large instances.

Chen et al. (2015) dealt with a JOBPRP in which splitting of customer orders is not allowed. The capacity of the picking device is limited by the total weight of items and a single order picker is available for processing customer orders. They aimed for the minimization of the total tardiness. Chen et al. (2015) designed a genetic algorithm in order to tackle the batching subproblem. In the genetic algorithm, an initial solution is generated by means of the earliest due date rule, i.e. customer orders are sorted in an order of non-descending due dates and batches are constructed based on this order. Chromosomes are divided into two gene segments. Alleles in the first segment give information about the size of a batch, while the second segment determines the sequence according to which orders are assigned to a batch. Two-point crossover operations are applied to the first gene segment and position-based crossover operations to the second segment. According to the mutation operation, two alleles from the same segment are exchanged. For the determination of the fitness value of a chromosome, the arising routing problems are solved by the application of an ACO approach. The integration of an ACO approach into a genetic algorithm results in a very time-consuming algorithm. Chen et al. (2015) dealt with small instances including up to 8 customer orders only and stated that computing time is a very critical issue even for such small problem instances.

Scholz et al. (2017) considered the same settings as Chen et al. (2015) but they measured the capacity in terms of the number of items. Furthermore, they dealt with multiple pickers, which makes the problem more complex since decisions regarding the assignment of batches to pickers have to be made as well. Scholz et al. (2017) proposed a variable neighborhood descent (VND) algorithm for the batching problem. For the generation of an initial solution, an earliest due date rule-based algorithm and a seed algorithm are applied, while the solution with the smaller objective function value is taken as initial solution. The VND includes six different neighborhood structures from which two structures alter the batch assignment and sequencing. The remaining four structures are shift and swap moves regarding the composition of the batches. Tours are constructed by means of two different routing algorithms.

Within the local search phases, the combined heuristic is applied as it generates tours of good quality within a very small amount of computing time (Roodbergen & de Koster, 2001b). Whenever a new best solution has been found, the LKH heuristic of Helsgaun (2000) is used in order to improve the tours, further reducing the total tardiness. In the results of the numerical experiments, Scholz et al. (2017) demonstrated that this approach can deal with very large instances. Problem instances with up to 200 orders and 5 pickers have been solved. The maximum computing time amounted to 1 hour.

5 Conclusion and outlook

The Picker Routing Problem (PRP) deals with the determination of the sequence according to which storage locations of requested items are to be visited and the identification of the corresponding path through the picking area of the warehouse. It is the most studied problem of all warehouse operations (Gu et al., 2007), which is not surprising as traveling consumes the major part of an order picker's working time (Tompkins et al., 2010) and routing order pickers is considered to be pivotal for an efficient organization of order picking operations.

The PRP is mainly characterized by the layout of the picking area, which is determined by the arrangement of the storage locations. The layout can be classified by the width of the picking aisles (standard, wide or narrow), the arrangement of picking and cross aisles (single-block, multi-block or non-conventional layout) as well as by the number of deposit locations and the number of storage locations to which an article may be assigned (single or multiple locations). The PRP in standard-aisle warehouses with a single depot and unique article locations has frequently been studied in the literature. Exact approaches are available to the PRP in conventional layouts with up to two blocks and for the PRP in fishbone and flying-V layouts. Furthermore, it has been shown that these approaches can easily be extended to the case of multiple deposit locations. The main research gap in the context of PRPs in standard-aisle warehouses can be found in the consideration of multiple storage locations per article. By assigning articles to multiple storage locations, tours become much more flexible, resulting in decreased travel distances. However, first, the benefits of assigning an article to multiple locations concerning the tour length have not been evaluated so far. Second, only one solution approach exists, which can deal with this feature. Thus, the development of an exact approach and the modification of routing strategies to the case of multiple article locations would represent a promising area for future research.

In wide picking aisles, additional movements have to be performed in the picking aisles in order to retrieve items from different sides of the picking aisle. Due to this fact, the corresponding PRP gets slightly more difficult to solve. However, it can be shown that exact approaches to PRPs in standard-aisle warehouses can be adapted easily to the case of wide aisles. This is not possible when dealing with narrow aisles where order pickers can neither pass nor overtake each other. PRPs with narrow aisles are much more complex to handle because routes of different pickers cannot be constructed independently anymore and waiting times caused by order pickers blocking each other have to be taken into account. Two heuristic approaches exist which can deal with such PRPs. However, the solution quality does not seem to be convincing. Thus, the main emphasis should be put on the development of exact and heuristic solution approaches to the PRP in narrow-aisle warehouses. By means of such algorithms, it could then be investigated under which conditions, narrow aisles should be used in order to maximize the space utilization in the picking area, and when it is inevitable to design a layout with standard aisles in order to keep the processing times at a reasonable level. Further research could also concentrate on the investigation of the performance of standard-aisle PRP approaches when being applied to PRPs with narrow aisles. Then, conclusions could be drawn about which routing strategy leads to least blocking situations.

The Order Batching Problem deals with the grouping of customer orders into batches and always arises simultaneously with the Picker Routing Problem. Thus, it is not surprising that the integrated solution of both problems has received much attention in the literature so far. The solution approaches to the Joint Order Batching and Picker Routing Problem (JOBPRP) have the same structure and are composed of a batching heuristic including a routing algorithm for determining the objective function value. Although all approaches have the same components, the approaches rarely rely on the same assumptions, making it almost impossible to compare the algorithms with respect to solution quality or computing time. In general, the assumptions concern the objective and the way how the capacity is measured. Distance-related and tardiness-related objectives are considered, where additional decisions have to be made concerning the assignment of batches to order pickers and the sequence of the batches assigned to a picker when dealing with tardiness-related objectives. Thus, solution approaches to the JOBPRP with a distance-related objective cannot be adapted straightforwardly to the case of tardiness-related objectives. In contrast, the assumption regarding the capacity measurement is not critical and could easily be changed in the respective algorithms. Therefore, it would be interesting to make all algorithms applicable to all three possible types of capacity measurements and then compare the performance of the algorithms.

All approaches to the JOBPRP deal with standard-aisle warehouses. As mentioned above, an adaption to warehouses with wide picking aisles would be quite easy. However, the consideration of the JOBPRP in narrow-aisle warehouses would be a very interesting topic for future research, although the resulting problem can be expected to be very challenging. Another extension of the JOBPRP can be found in the integration of multiple article locations. Compared to the consideration of multiple article locations in the context of the PRP, the benefits could be even higher when dealing with the JOBPRP, since more combinations of orders represent promising batches if it can be chosen between several article locations. Furthermore, in case of narrow aisles, assigning articles to multiple locations allows for more flexible tours and can be expected to significantly decrease the waiting times caused by blocking situations.

In this paper, operations performed in the picking area have been considered. After the items have been retrieved they are transported to the shipping area to be then delivered to the customers. Further research could also concentrate on the integration of picking and shipping operations. The problem of delivering the items to the customer locations is typically formulated as a Vehicle Routing Problem with Time Windows because customers usually give a certain time interval when they are able to take the delivery of the goods (Schmid et al., 2013). A solution to the shipping problem consists of a set of tours with different start dates, while the start date of a tour is dependent on the point in time when the items of all orders allocated to the tour have been retrieved completely. Thus, in order to guarantee the tours to start in time, it is pivotal to coordinate picking and shipping operations.

References

Burkard, R.; Deneko, V. G; van der Veen, J. A. A. & Woeginger, G. J. (1998): Well-Solvable Special Cases of the Traveling Salesman Problem: A Survey. *SIAM Review* 40, 496-546.

Çelik, M. & Süral, H. (2014): Order Picking under Random and Turnover-Based Storage Policies in Fishbone Aisle Warehouses. *IIE Transactions* 46, 283-300.

Chen, F.; Wang, H.; Qi, C. & Xie, Y. (2013): An Ant Colony Optimization Routing Algorithm for Two Order Pickers with Congestion Consideration. *Computers & Industrial Engineering* 66, 77-85.

Chen, T.-L.; Cheng, C.-Y.; Chen, Y.-Y. & Chan, L.-K. (2015): An Efficient Hybrid Algorithm for Integrated Order Batching, Sequencing and Routing Problem. *International Journal of Production Economics* 159, 158-167.

Chen, F.; Wang, H.; Xie, Y. & Qi, C. (2016): An ACO-Based Online Routing Method for Multiple Order Pickers with Congestion Consideration in Warehouse. *Journal of Intelligent Manufacturing* 27, 389-408.

Cheng, C.-Y.; Chen, Y.-Y.; Chen, T.-L. & Yoo, J. J.-W. (2015): Using a Hybrid Approach Based on the Particle Swarm Optimization and Ant Colony Optimization to Solve a Joint Order Batching and Picker Routing Problem. *International Journal of Production Economics* 170, 805-814.

Clarke, G. & Wright, J. W. (1964): Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research* 12, 568-581.

Coyle, J. J.; Bardi, E. J. & Langley, C. J. (1996): The Management of Business Logistics. 6th ed., West Publishing Company: St. Paul.

Daniels, R. L.; Rummel, J. L. & Schantz, R. (1998): A Model for Warehouse Order Picking. *European Journal of Operational Research* 105, 1-17.

de Koster, R. & van der Poort, E. (1998): Routing Orderpickers in a Warehouse: A Comparison between Optimal and Heuristic Solutions. *IIE Transactions* 30, 469-480.

de Koster, R.; Le-Duc, T. & Roodbergen, K. J. (2007): Design and Control of Warehouse Order Picking: A Literature Review. *Science Direct* 182, 481-501.

de Koster (2008): Warehouse Assessment in a Single Tour. *Facility Logistics: Approaches and Solutions to Next Generation Challenges*, Lahmar, M. (ed.), 39-60, Taylor & Francis Group: New York.

Frazelle, E. (2002): World-Class Warehouse and Material Handling. McGrawHill: New York.

Gademann, N. & van de Velde, S. (2005): Order Batching to Minimize Total Travel Time. *IIE Transactions* 37, 63-75.

Goetschalckx, M. & Ratliff, H. D. (1988): Order Picking in an Aisle. *IIE Transactions* 20, 53-62.

Grosse, E. H.; Glock, C. H. & Ballester-Ripoll, R. (2014): A Simulated Annealing Approach for the Joint Order Batching and Order Picker Routing Problem with Weight Restrictions. *International Journal of Operations and Quantitative Management* 20, 65-83.

Gu, J.; Goetschalckx, M. & McGinnis, L. F. (2007): Research on Warehouse Operation: A Comprehensive Review. *European Journal of Operational Research* 177, 1-21.

Gue, K. R. & Meller, R. D. (2009): Aisle Configuration for Unit-Load Warehouses. *IIE Transactions* 41, 171-182.

Hall, R. W. (1993): Distance Approximations for Routing Manual Pickers in a Warehouse. *IIE Transactions* 25, 76-87.

Helsgaun, K. (2000): An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic. *European Journal of Operational Research* 126, 106-130.

Henn, S.; Koch, S.; Dörner, K.; Strauss, C. & Wäscher, G. (2010): Metaheuristics for the Order Batching Problem in Manual Order Picking Systems. *BuR - Business Research* 3, 82-105.

Henn, S.; Koch, S. & Wäscher, G. (2012): Order Batching in Order Picking Warehouses: A Survey of Solution Approaches. *Warehousing in the Global Supply Chain: Advanced Models, Tools and Applications for Storage Systems*, Manzini, R. (ed.), 105-137, Springer: London.

Henn, S. & Schmid, V. (2013): Metaheuristics for Order Batching and Sequencing in Manual Order Picking Systems. *Computers & Industrial Engineering* 66, 338-351.

Jarvis, J. M. & McDowell, E. D. (1991): Optimal Product Layout in an Order Picking Warehouse. *IIE Transactions* 23, 93-102.

Kulak, O.; Sahin, Y. & Taner, M. E. (2012): Joint Order Batching and Picker Routing in Single and Multiple-Cross-Aisle Warehouses Using Cluster-Based Tabu Search Algorithms. *Flexible Services and Manufacturing Journal* 24, 52-80.

Lin,C.-C.; Kang, J.-R.; Hou, C.-C. & Cheng, C.-Y. (2016): Joint Order Batching and Picker Manhattan Routing Problem. *Computers & Industrial Engineering* 95, 164-174.

Matusiak, M.; De Koster, R.; Kroon, L. & Saarinen, J. (2014): A Fast Simulated Annealing Method for Batching Precedence-Constrained Customer Orders in a Warehouse. *European Journal of Operational Research* 236, 968-977.

Parikh, P. J. & Meller, R. D. (2009): Estimating Picker Blocking in Wide-Aisle Order Picking Systems. *IIE Transactions* 41, 232-246.

Petersen, C. G. (1997): An Evaluation of Order Picking Routeing Policies. *International Journal of Operations and Production Management* 17, 1098-1111.

Petersen, C. G. & Schmenner, R. W. (1999): An Evaluation of Routing and Volume-Based Storage Policies in an Order Picking Operation. *Decision Science* 30, 481-501.

Psaraftis, H. (1980): Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem. *Transportation Science*, 14, 130-154.

Ratliff, H. D. & Rosenthal, A. R. (1983): Order-Picking in a Rectangular Warehouse: A Solvable Case of the Traveling Salesman Problem. *Operations Research* 31, 507-521.

Roodbergen, K. J. (2001): Layout and Routing Methods for Warehouses. Trial: Rotterdam.

Roodbergen, K. J. & de Koster, R. (2001a): Routing Order Pickers in a Warehouse with a Middle Aisle. *European Journal of Operational Research* 133, 32-43.

Roodbergen, K. J. & de Koster, R. (2001b): Routing Methods for Warehouses with Multiple Cross Aisles. *International Journal of Production Research* 39, 1865-1883.

Roodbergen, K. J.; Sharp, G. P. & Vis, I. F. A. (2008): Designing the Layout Structure of Manual Order-Picking Areas in Warehouses. *IIE Transactions* 40, 1032-1045.

Rouwenhorst, B.; Reuter, B.; Stockrahm, V.; van Houtum, G.J.; Mantel, R.J. & Zijm, W.H.M (2000): Warehouse Design and Control: Framework and Literature Review. *European Journal of Operational Research* 121, 515-533.

Schmid, V.; Doerner, K. F & Laporte, G. (2016): Rich Routing Problems Arising in Supply Chain Management. *European Journal of Operational Research* 224, 435-448.

Scholz, A. (2016): An Exact Solution Approach to the Single-Picker Routing Problem in Warehouses with an Arbitrary Block Layout. Working Paper No. 6/2016, Faculty of Economics and Management, Otto-von-Guericke University Magdeburg.

Scholz, A.; Henn, S.; Stuhlmann, M. & Wäscher, G. (2016): A New Mathematical Programming Formulation for the Single-Picker Routing Problem. *European Journal of Operational Research* 253, 68-84.

Scholz, A.; Schubert, D. & Wäscher, G. (2017): Order Picking with Multiple Pickers and Due Dates Simultaneous Solution of Order Batching, Batch Assignment and Sequencing, and Picker Routing Problems. *European Journal of Operational Research*, DOI: 10.1016/j.ejor.2017.04.038.

Scholz, A. & Wäscher, G. (2017): Order Batching and Picker Routing in Manual Order Picking Systems: The Benefits of Integrated Routing. *Central European Journal of Operations Research* 25, 491-520.

Theys, C.; Bräysy, O.; Dullaert, W. & Raa, B. (2010): Using a TSP Heuristic for Routing Order Pickers in Warehouses. *European Journal of Operational Research* 200, 755-763.

Tompkins, J. A.; White, J. A.; Bozer, Y. A. & Tanchoco, J. M. A. (2010): Facilities Planning. 4th edition, John Wiley & Sons, New Jersey.

Tsai, C.-Y.; Liou, J. J. H. & Huang, T.-M. (2008): Using a Multiple-GA Method to Solve the Batch Picking Problem: Considering Travel Distance and Order Due Time. *International Journal of Production Research* 46, 6533-6555.

Vaughan, T. S. & Petersen, C. G. (1999): The Effect of Warehouse Cross Aisles on Order Picking Efficiency. *International Journal of Production Research* 37, 881-897.

Wäscher, G. (2004): Order Picking: A Survey of Planning Problems and Methods. *Supply Chain Management and Reverse Logistics*, Dyckhoff , H.; Lackes, R. & Reese, J. (eds.), 323-347, Springer: Berlin.

Won, J. & Olafsson, S. (2005): Joint Order Batching and Order Picking in Warehouse Operations. *International Journal of Production Research* 43, 1427-1442.

Part III:

Picker Routing in Standard-Aisle, Single-Block Warehouses

European Journal of Operational Research 253 (2016) 68–84

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Production, Manufacturing and Logistics

A new mathematical programming formulation for the Single-Picker Routing Problem

André Scholz[a,*], Sebastian Henn[a], Meike Stuhlmann[a], Gerhard Wäscher[a,b]

[a] Faculty of Economics and Management, Otto-von-Guericke University Magdeburg, 39106 Magdeburg, Germany
[b] School of Mechanical, Electronic and Control Engineering, Beijing Jiaotong University, 100044 Beijing, China

ARTICLE INFO

Article history:
Received 30 April 2015
Accepted 10 February 2016
Available online 17 February 2016

Keywords:
Traveling Salesman
Order picking
Picker Routing

ABSTRACT

The Single-Picker Routing Problem deals with the determination of the sequence according to which article locations have to be visited in a distribution warehouse and the identification of the corresponding paths which have to be traveled by human operators (order pickers) in order to collect a set of items requested by internal or external customers. The Single-Picker Routing Problem (SPRP) represents a special case of the classic Traveling Salesman Problem (TSP) and, therefore, can also be modeled as a TSP. Standard TSP formulations applied to the SPRP, however, neglect that in distribution warehouses article locations are arranged in a specifically structured way. When arranged according to a block layout, articles are located in parallel picking aisles, and order pickers can only change over to another picking aisle at certain positions by means of so-called cross aisles. In this paper, for the first time a mathematical programming formulation is proposed which takes into account this specific property. By means of extensive numerical experiments it is shown that the proposed formulation is superior to standard TSP formulations.

1. Introduction

The Traveling Salesman Problem (TSP) is one of the most extensively studied problems in combinatorial optimization (Rego, Gamboa, Glover, & Osterman, 2011). It can be described as the problem of finding a least-weight Hamiltonian cycle (in short: a tour) in a complete edge-weighted graph (Glover & Punnen, 1997). The TSP arises in many different practical contexts. It is of prime importance for practical applications in engineering, management, health care and many other areas. For detailed reviews of applications of the TSP in practice we refer to Lenstra and Rinnooy Kan (1975), Matai, Singh, and Mittal (2010) and Filip and Otakar (2011).

In this paper, we will deal with an application of the TSP that arises in distribution warehouse management. As for its core function, order picking, items have to be retrieved from the warehouse in order to satisfy a given demand from customers (Petersen & Schmenner, 1999; Wäscher, 2004). In picker-to-part systems, human operators (order pickers) travel through the warehouse, collecting the requested items at their storage locations. This gives rise to the so-called *Single-Picker Routing Problem* (SPRP) which includes the determination of the sequence in which the locations have to be visited. It can be interpreted as a special case of the (classic) TSP or the Steiner TSP (Burkard, Deneko, van der Veen, & Woeginger, 1998).

For the (classic) TSP several mathematical programming formulations have been proposed in the literature (Padberg & Sung, 1991). These formulations and also formulations for the Steiner TSP, however, seem not to be appropriate for the SPRP since they ignore the special structure of the latter. In this paper, the first mathematical formulation for the SPRP will be introduced that takes into account specific properties of optimal solutions of this problem. In comparison to the more general modeling approaches mentioned above, the proposed formulation results in a substantial reduction of the number of variables and constraints. In particular, it will be shown that the size of the formulation is independent of the number of locations where items have to be picked from. By means of extensive numerical experiments it will be demonstrated that the proposed formulation for the SPRP is superior to standard TSP and Steiner TSP formulations.

The remainder of this paper is organized as follows: In the following section we introduce the SPRP and review the related literature. In Section 3, we present three well-known general mathematical programming formulations for the TSP as well as a formulation for the Steiner TSP provided by Letchford, Nasiri, and Theis (2013). Since the size of the models will be an important criterion for the analysis and evaluation of the different

* Corresponding author. Tel.: +49 3916711841.
E-mail address: andre.scholz@ovgu.de (A. Scholz).

http://dx.doi.org/10.1016/j.ejor.2016.02.018
0377-2217/

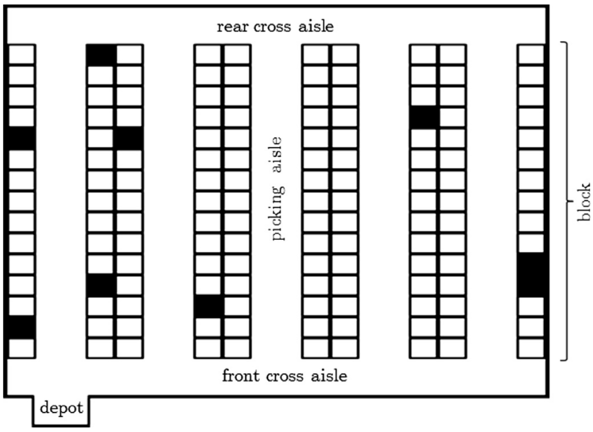


Fig. 1. Single-block layout.

formulations, we concentrate on compact formulations for the TSP here, i.e. formulations which only require a polynomial number of variables and constraints. In Section 4, the central part of this paper, we develop the newly proposed formulation for the SPRP. First, we describe a graph and its construction on which the formulation is based. Then we represent the graph in a mathematical programming model and exemplify its core elements. In order to analyze and evaluate the different formulations, extensive numerical experiments have been carried out in which a wide range of problem instances have been solved by a state-of-the-art commercial IP solver. The design of the experiments and the results obtained from them are presented in Section 5.

For the sake of simplicity of exposition, we restrain the presentation of the newly-proposed model in Section 4 and the numerical experiments in Section 5 to the so-called single-block layout which represents a standard arrangement of storage locations in distribution warehouses. In Section 6, we extend our view and demonstrate how the graph and the resulting model formulation can be modified in order to represent more complex layouts and other aspects of order picking encountered in practice. In Section 7, the paper concludes with a summary and an outlook on future research areas. An Appendix includes a complete and detailed model formulation for the SPRP.

2. The Single-Picker Routing Problem

2.1. Problem description

In picker-to-part order picking systems, the arrangement of storage locations typically follows a *block layout* (Roodbergen, 2001) in which *picking aisles* run in parallel to each other. Articles are stored in and picked from racks on both sides of these picking aisles. *Cross aisles* can be used to proceed from one picking aisle to another. They do not contain any storage locations. A section between two adjacent cross aisles establishes a so-called block. In the following, we will focus on a single-block layout, i.e. only two cross aisles exist, one at the front and one at the rear of the warehouse (see Fig. 1).

The black rectangles in Fig. 1 give an example of locations from which items have to be collected (*pick locations*) in order to satisfy a certain demand from (external or internal) customers. This is done by a so-called *order picker*, a human operator who completes a tour through the warehouse, i.e. he or she starts from a *depot*, proceeds to the pick locations, retrieves the requested items, and finally returns to the depot where the picked items are deposited.

Due to the high proportion of time-consuming manual tasks, order picking is looked upon as the most labor cost-intensive warehouse function (Tompkins, White, Bozer, & Tanchoco, 2010). Consequently, the minimization of picking times is of vital importance for the efficient control of picking operations. The *total order picking time*, i.e. the time spent by an order picker to collect all items of a picking order, can be divided into (Tompkins et al., 2010) the *setup time* (the time for preparing the tour through the warehouse), the *travel time* (the time needed to travel to, from, and between the pick locations), the *search time* (the time needed at the article locations for the identification of the items that have to be retrieved), and the *pick time* (the time actually needed for retrieving the items from the respective article locations). Among these components, the travel time consumes the major proportion of the total order picking time. It also represents the only variable part while the remaining components (setup time, search time, pick times) can be considered to be constants (Bozer & Kile, 2008; Henn, Koch, Dörner, Strauss, & Wäscher, 2010). The travel time is determined by the total length of the picker tour (Jarvis & McDowell, 1991) which, again, is dependent on the sequence according to which the items have to be picked. Assuming that the order picker moves at a constant velocity, the minimization of the *total length of the picker tour* becomes equivalent to the minimization of the travel time. Therefore, the SPRP can be defined as follows: Given a set of items to be picked from known storage locations, in which sequence should the locations be visited such that the total length of the corresponding picker tour is minimized?

2.2. Related literature

The SPRP can easily be interpreted as a TSP in which the vertices of the corresponding graph are defined by the location of the depot and the locations of the items to be picked. Therefore, general TSP formulations may also be used to model the SPRP. The first mathematical programming formulation for the TSP can be attributed to Dantzig, Fulkerson, and Johnson (1954). It includes one binary variable per edge indicating whether an edge is contained in the tour or not. However, this formulation requires, like several other formulations (Gouveia & Pires, 2001), an exponential number of constraints.

For an explicit representation of large TSPs, compact formulations appear to be more appropriate which only require a polynomial number of variables and constraints. A variety of such formulations have been proposed in the literature (Öncan, Altinel, & Laporte, 2009), of which the formulation of Miller, Tucker, and Zemlin (1960), the single-commodity flow formulation of Gavish and Graves (1978) and the multi-commodity flow formulation of Claus (1984) probably are the most prominent ones. They will be explained in greater detail in Section 3, since we will compare our modeling approach to these formulations.

Burkard et al. (1998) pointed out that the SPRP can also be formulated as a Steiner TSP, which can be defined as follows: Let $G = (V, E)$ be a graph with a set of vertices V and a set of edges E. Furthermore, let a sequence $v_1, e_1, v_2, \ldots, v_k, e_k, v_{k+1}$ $(v_i \in V,$ $e_i = (v_i, v_{i+1}),$ $i = 1, \ldots, k,$ $k \geq 0)$ with $e_i \neq e_j$ for $i \neq j$ and $v_1 = v_{k+1}$ be called a closed walk. Let P be a subset of V. The elements of $V \backslash P$ are called Steiner points. A Steiner tour is then defined as a closed walk in which each vertex of P is visited at least once. The Steiner points do not have to be visited. Then the Steiner TSP consists of finding a Steiner tour of minimal length within G (Burkard et al., 1998).

As for the SPRP (see Fig. 2), the set P is composed of the pick locations and the depot (black vertices), and the Steiner points are the intersections between the picking aisles and the cross aisles (white vertices). The distance between any pair of vertices is the length of the shortest path between the two vertices. By

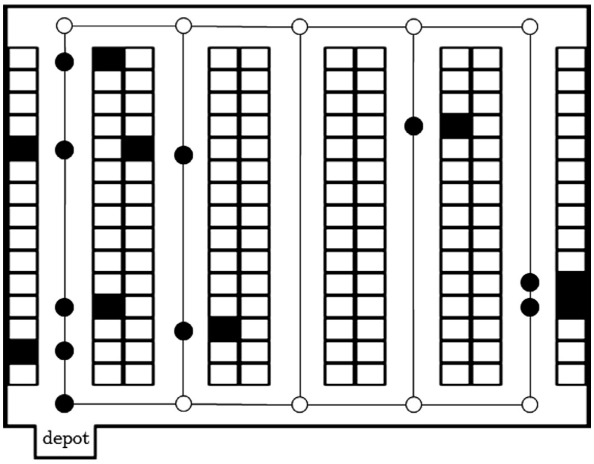


Fig. 2. Illustration of a Steiner TSP.

definition, a Steiner tour then has to include the depot and the storage locations of all items to be picked. Visiting these locations more than once is permitted. Multiple visits may occur, e.g., when a picking aisle is entered and left via the same cross aisle. On the other hand, visits of some of the white vertices may be skipped because two points exist for each picking aisle where an aisle can be entered or left, and it is possible to use one vertex twice and skip the other one.

Letchford et al. (2013) developed three compact formulations for the Steiner TSP. These formulations are advantageous in comparison to general TSP formulations if the number of Steiner points is large compared to the total number of vertices (Letchford et al., 2013). For problem instances in which less than 100 vertices (including both Steiner points and locations to be visited) have to be considered, the authors proposed the application of a Steiner TSP formulation which is derived from the multi-commodity flow model of Claus (1984). Since instances of the SPRP from practice will usually result in a Steiner TSP with less than 100 vertices, we will focus on this formulation. It will be explained in greater detail in Section 3.

The SPRP in a single-block or a two-block layout can be solved efficiently by problem-specific solution methods. Ratliff and Rosenthal (1983) introduced an optimal algorithm for the single-block case which is based on dynamic programming and solves the problem in $O(m+n)$ time, where m is the number of picking aisles and n is the number of pick locations. Roodbergen and de Koster (2001a) modified the algorithm for the two-block case. However, it is very difficult to further extend this algorithm to layouts with three or more blocks and up to now no efficient algorithm exists which can deal with an arbitrary number of blocks (Roodbergen, 2001). Our approach is much more flexible with respect to the extension to different layouts which we consider as a main advantage of our approach. In particular, it allows for modeling a block layout with an arbitrary number of blocks and considering the case of decentralized depositing in which items can be deposited at the end of each picking aisle. Furthermore, our proposed model formulation can easily be modified to find simple tours through the warehouse which will be demonstrated in Section 6.

In practice, in order to solve the SPRP, usually simple heuristics, so-called *routing strategies*, are applied (Roodbergen, 2001). Solutions from such routing strategies can be memorized and executed easily, and they reduce the risk of missing an item that should be picked. The simplest routing strategies are the S-shape, the return and the largest gap strategies (Gu, Goetschalckx, & McGinnis, 2007; de Koster, Le-Duc, & Roodbergen, 2007). The *S-shape strategy* provides solutions in which the order picker enters a picking aisle and traverses it completely if at least one required item is located in that aisle. Then he proceeds to the next aisle from which items have to be picked. An exception would only be a situation in which the order picker is positioned in the front cross aisle, facing the last picking aisle from which items have to be picked. He would then return to the front cross aisle after having picked the most-distantly located item in that aisle. Solutions from the *return strategy* provide tours in which each picking aisle containing at least one requested item is entered from the front cross aisle. The picker proceeds to the farthest pick location and returns to the front cross aisle with the picked items. The *largest-gap strategy* gives solutions in which the order picker completely traverses the first and the last aisle containing a requested item. All other aisles containing at least one required item are entered from the front and from the rear cross aisle in such a way that the non-traversed distance between two adjacent pick locations or a pick location and the adjacent cross aisle is maximal. The *combined strategy* (Roodbergen & de Koster, 2001b) integrates elements of the S-shape and return strategy. Aisles may be traversed entirely or may be entered and left via the same cross aisle. Solutions are generated by means of dynamic programming.

The performance of the routing strategies is dependent on the problem characteristics (number of picking aisles, number of locations per aisle, position of the depot, number of requested items). Moreover, also the policy according to which items are assigned to different locations has a significant impact on the tour lengths provided by the heuristics. Elements of these routing strategies will be contained in the new model formulation for the SPRP presented in this paper.

3. General TSP and Steiner TSP formulations

For a model formulation of the TSP usually a complete graph $G^{TSP} = (V, A)$ is assumed. In case of the SPRP, the set of vertices $V = \{0, \ldots, n\}$ contains the depot (vertex 0) and all storage locations where items have to be picked. The set of arcs is defined as $A = \{(p, q) \mid p, q \in V, p \neq q\}$ and for each arc a distance c_{pq} can be calculated according to the respective layout.

Based on this graph, we will focus on three model formulations which require a polynomial number of both variables and constraints. These formulations, which have also been chosen by Letchford et al. (2013) as a basis for their analysis of Steiner TSP formulations, differ with respect to two characteristics, namely the number of variables and constraints on the one hand and the quality of the lower bound which can be obtained by solving the LP relaxation on the other hand.

We complete this section of model formulations by presenting the Steiner TSP formulation that was proposed by Letchford et al. (2013) and derived from the multi-commodity flow model of Claus (1984).

3.1. Formulation of Miller, Tucker and Zemlin

The formulation proposed by Miller et al. (1960) uses the following variables:

$$x_{pq} = \begin{cases} 1, & \text{if arc } (p,q) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad (p,q) \in A;$$

h_p : position of vertex p in the tour, $p \in V \setminus \{0\}$.

Then, the TSP can be represented as follows:

$$\min \sum_{(p,q)\in A} c_{pq} \cdot x_{pq} \qquad (1)$$

$$\sum_{p\in V} x_{pq} = 1 \qquad \forall\, q \in V \tag{2}$$

$$\sum_{q\in V} x_{pq} = 1 \qquad \forall\, p \in V \tag{3}$$

$$h_p - h_q + (n+1)x_{pq} \le n \qquad \forall\,(p,q) \in A\colon\ p, q \neq 0 \tag{4}$$

$$x_{pq} \in \{0,1\} \qquad \forall\,(p,q) \in A \tag{5}$$

$$h_p \ge 0 \qquad \forall\, p \in V\backslash\{0\} \tag{6}$$

The objective function minimizes the total cost of the tour, i.e. as for the SPRP the total length of the picker tour. Constraints (2) and (3) guarantee that each vertex is visited exactly once. Conditions (4) exclude subtours by ensuring that the position assigned to vertex p in a tour is smaller than the position assigned to vertex q if arc (p, q) is used. This formulation requires only $O(n^2)$ variables and constraints. However, the solution of its LP relaxation provides an extremely weak lower bound (Padberg & Sung, 1991).

3.2. Formulation of Gavish and Graves

A formulation based on the flow of a single commodity type has been introduced by Gavish and Graves (1978). At the start of the tour, n units of the commodity are available. While the tour is being executed, these units have to be delivered to the vertices when they are visited. Each vertex requires the shipment of exactly one unit.

For this model, additional non-negative variables are introduced describing the flow on arc $(p, q) \in A$:

g_{pq} : number of units of the commodity passed on directly from vertex p to q, $(p,q) \in A : q \neq 0$.

Based on this definition and on the above-described considerations, the following model formulation can be introduced:

$$\min \sum_{(p,q)\in A} c_{pq} \cdot x_{pq} \tag{7}$$

$$\sum_{p\in V} x_{pq} = 1 \qquad \forall\, q \in V \tag{8}$$

$$\sum_{q\in V} x_{pq} = 1 \qquad \forall\, p \in V \tag{9}$$

$$\sum_{q\in V} g_{qp} - \sum_{q\in V\backslash\{0\}} g_{pq} = 1 \qquad \forall\, p \in V\backslash\{0\} \tag{10}$$

$$g_{pq} \le n x_{pq} \qquad \forall\,(p,q) \in A\colon\ q \neq 0 \tag{11}$$

$$x_{pq} \in \{0,1\} \qquad \forall\,(p,q) \in A \tag{12}$$

$$g_{pq} \ge 0 \qquad \forall\,(p,q) \in A\colon\ q \neq 0 \tag{13}$$

Constraints (10) ensure that exactly one unit of the commodity is delivered to each vertex $p \in V\backslash\{0\}$ while constraints (11) guarantee the flow to be zero along arcs not included in the tour. This formulation includes $O(n^2)$ variables and constraints. Padberg and Sung (1991) have shown that its LP relaxation leads to a stronger lower bound than the formulation of Miller et al. (1960).

3.3. Formulation of Claus

Another formulation that was proposed by Claus (1984) uses multi-commodity flows in order to prohibit subtours. Here, n different commodities have to be delivered. At the beginning of the tour, one unit of each commodity is available. Each vertex is meant to receive exactly one of these commodities.

For the presentation of this model formulation, the following additional variables are introduced:

w^k_{pq} : number of units of commodity k passed on directly from vertex p to q, $(p,q) \in A, k \in V \setminus \{0\}$.

The TSP can then be represented as follows:

$$\min \sum_{(p,q)\in A} c_{pq} \cdot x_{pq} \tag{14}$$

$$\sum_{p\in V} x_{pq} = 1 \qquad \forall\, q \in V \tag{15}$$

$$\sum_{q\in V} x_{pq} = 1 \qquad \forall\, p \in V \tag{16}$$

$$\sum_{q\in V\backslash\{0\}} w^k_{1q} - \sum_{q\in V\backslash\{0\}} w^k_{q1} = -1 \qquad \forall\, k \in V\backslash\{0\} \tag{17}$$

$$\sum_{q\in V} w^p_{pq} - \sum_{q\in V} w^p_{qp} = 1 \qquad \forall\, p \in V\backslash\{0\} \tag{18}$$

$$\sum_{q\in V} w^k_{pq} - \sum_{q\in V} w^k_{qp} = 0 \qquad \forall\, p, k \in V\backslash\{0\}\colon\ p \neq k \tag{19}$$

$$w^k_{pq} \le x_{pq} \qquad \forall\,(p,q) \in A,\ k \in V\backslash\{0\} \tag{20}$$

$$x_{pq} \in \{0,1\} \qquad \forall\,(p,q) \in A \tag{21}$$

$$w^k_{pq} \ge 0 \qquad \forall\,(p,q) \in A,\ k \in V\backslash\{0\} \tag{22}$$

Constraints (17) guarantee that each commodity leaves the depot and is delivered to a vertex. Constraints (18) ensure that each vertex receives exactly one commodity. Constraints (19) ensure that a commodity leaves a vertex that is not its final destination. This model formulation requires $O(n^3)$ variables and constraints. The solution of its LP relaxation leads to the strongest lower bound of the three formulations considered here (Padberg & Sung, 1991).

3.4. Formulation of Letchford, Nasiri and Theis

In a block layout, the order picker cannot proceed directly from one location of a requested item to another one if these items are located in different picking aisles. Instead, a cross aisle has to be used for switching over from one picking aisle to the other. This aspect is neglected in general TSP formulations since a complete graph is assumed. It can be taken into account explicitly, though, in Steiner TSP formulations. The following formulation for the Steiner TSP was provided by Letchford et al. (2013) and has been derived from the general multi-commodity flow TSP formulation of Claus (1984).

Since the set of arcs in this formulation differs from the arc set of the general TSP formulation, we use $\tilde{A}$ to denote the set of arcs of a Steiner TSP. As in Section 2, the set of Steiner points is denoted by P. The formulation for the Steiner TSP then is as follows:

$$\min \sum_{(p,q)\in \tilde{A}} c_{pq} \cdot x_{pq} \tag{23}$$

$$\sum_{\substack{q\in V:\\ (p,q)\in \tilde{A}}} x_{pq} \ge 1 \qquad \forall\, p \in V\backslash P \tag{24}$$

$$\sum_{\substack{q\in V:\\ (p,q)\in \tilde{A}}} x_{pq} - \sum_{\substack{q\in V:\\ (q,p)\in \tilde{A}}} x_{qp} = 0 \qquad \forall\, p \in V \tag{25}$$

$$\sum_{\substack{q\in V:\\ (q,1)\in \tilde{A}}} w^k_{q1} - \sum_{\substack{q\in V:\\ (1,q)\in \tilde{A}}} w^k_{1q} = -1 \qquad \forall\, k \in V\backslash(P \cup \{0\}) \tag{26}$$

$$\sum_{\substack{q\in V:\\ (q,k)\in \tilde{A}}} w^k_{qk} - \sum_{\substack{q\in V:\\ (k,q)\in \tilde{A}}} w^k_{kq} = 1 \qquad \forall\, k \in V\backslash(P \cup \{0\}) \tag{27}$$

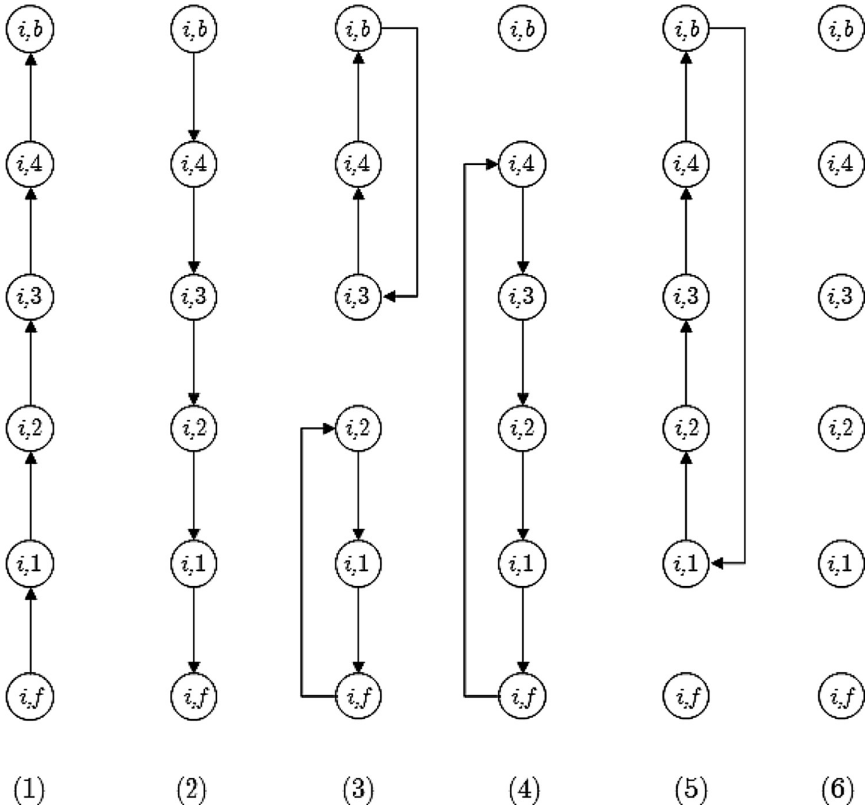


Fig. 3. Movements within a picking aisle permitted for an optimal solution.

$$\sum_{\substack{q \in V: \\ (p,q) \in \tilde{A}}} w_{pq}^{k} - \sum_{\substack{q \in V: \\ (q,p) \in \tilde{A}}} w_{qp}^{k} = 0 \quad \forall\, p \in V \backslash \{0\},\ k \in V \backslash (P \cup \{0, p\}) \tag{28}$$

$$w_{pq}^{k} \leq x_{pq} \quad \forall\, (p, q) \in \tilde{A},\ k \in V \backslash (P \cup \{0\}) \tag{29}$$

$$x_{pq} \in \{0, 1\} \quad \forall\, (p, q) \in \tilde{A} \tag{30}$$

$$w_{pq}^{k} \geq 0 \quad \forall\, (p, q) \in \tilde{A},\ k \in V \backslash (P \cup \{0\}) \tag{31}$$

Constraints (24) ensure that each vertex not corresponding to a Steiner point is visited at least once, while (25) guarantee that the indegree of each vertex is equal to its outdegree. Constraints (26)–(29) correspond to the multi commodity flow constraints ((17)–(20)) in the formulation of Claus (1984).

In general, the formulation of Letchford et al. (2013) for the Steiner TSP requires $O(|P||E|)$ variables and constraints. When it is being used for representing a SPRP, the location of the depot and the pick locations have to be taken as points which have to be visited ($|P| = n + 1$) while the points where a picking aisle can be entered or left represent Steiner points. Assuming the picking area following a block layout, $|E| = O(n \cdot m)$ arcs are needed to represent the SPRP as a Steiner TSP. This results in a model formulation which requires $O(n^2 \cdot m)$ variables and constraints, i.e. its size is dependent on both the number of requested items and the number of picking aisles. However, especially for a large number of storage locations to be visited, it requires significantly fewer variables and constraints than the formulation of Claus (1984) on which it is based.

Each of these four formulations can be used to model and solve the SPRP. However, the number of variables and constraints and, therefore, the computing times needed to solve the problems may grow quite large when the number of pick locations is increased. In order to find a model formulation of smaller size, it is necessary to change over from general TSP or Steiner TSP formulations to a more problem-specific formulation for the SPRP.

4. A new, problem-specific formulation for the SPRP

Tours taken by order pickers in warehouses in which the storage locations are arranged according to a block layout exhibit a specific structure. It results from the fact that cross aisles have to be used for switching over from one picking aisle to another. Also movements within picking aisles are rather restricted. Both properties are not explicitly considered by general TSP formulations.

In this section, we introduce a new graph for the SPRP that takes into account these properties and we show that the respective number of vertices and arcs is not dependent on the number of pick locations. Then, a TSP formulation is applied to this graph in order to provide a model for the problem.

4.1. Graph construction

A representation of the SPRP in a single-block layout must not necessarily be based on a complete graph with arcs between each pair of vertices. Ratliff and Rosenthal (1983) have shown that for the generation of an optimal tour only six different paths need to be considered which allow for visiting all pick locations of a picking aisles (see Fig. 3).

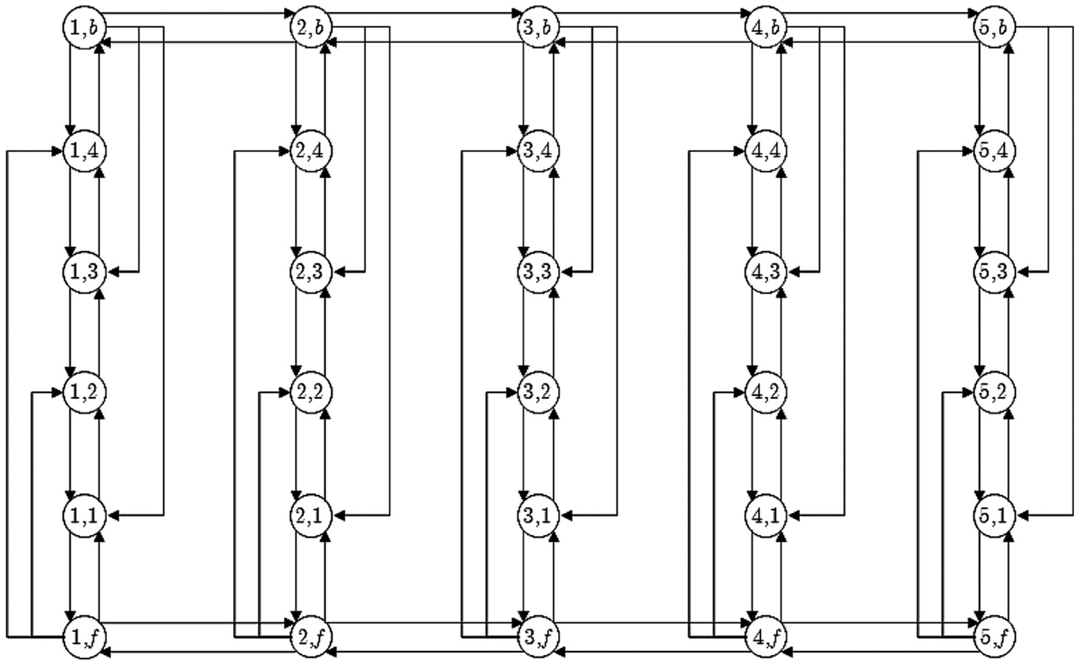


Fig. 4. A graph for a SPRP in a single-block layout with five picking aisles.

(1) The order picker enters the picking aisle from the front cross aisle, visits all pick locations sequentially, and exits the aisle at the rear cross aisle.
(2) The order picker enters the picking aisle from the rear cross aisle, visits all pick locations sequentially, and exits the aisle at the front cross aisle.
(3) The order picker enters and exits the picking aisle twice, one time from and back to the front cross aisle and another time from and back to the rear cross aisle. In both cases he returns to the cross aisle from where the picking aisle was entered. The return point is defined by the largest gap between two adjacent pick locations or between a pick location and the adjacent cross aisle.
(4) The order picker enters and leaves the aisle at the front cross aisle. The return point is defined by the pick location which corresponds to the largest distance from the front cross aisle.
(5) Likewise, the order picker enters and leaves the aisle at the rear cross aisle. The return point is defined by the pick location which corresponds to the largest distance from the rear cross aisle.
(6) The picking aisle is not entered at all since no requested item is located in that aisle.

As a consequence, instead of considering all pick locations, it is sufficient that each picking aisle i is only be represented by six points, namely the points which are defined by

(a) the intersection between the picking aisle i and the front cross aisle and the intersection between the picking aisle i and the rear cross aisle (vertices $[i, f]$ and $[i, b]$),
(b) the two pick locations defining the largest gap (vertices $[i, 2]$ and $[i, 3]$) and
(c) the first and the last storage location where an item has to be picked (vertices $[i, 1]$ and $[i, 4]$).

If less than four pick locations have to be visited in an aisle, points need to be duplicated in order to obtain the required number. In case that items have to be picked from one location only, that location will be represented by $[i, 1]$, $[i, 2]$, $[i, 3]$ and $[i, 4]$. In case items have to be picked from two different locations then $[i, 1]$ and $[i, 2]$ ($[i, 3]$ and $[i, 4]$) are identical. In case of three different pick locations that have to be visited, the pair of locations defining the largest gap has to be determined. If the gap exists between the two pick locations nearest to the front cross aisle, then $[i, 1]$ and $[i, 2]$ are identical; otherwise $[i, 3]$ and $[i, 4]$ are identical.

Based on these considerations, a graph representing the SPRP in a single-block layout can be constructed by introducing the six vertices for each picking aisle and choosing the edges that result from the options according to which pick locations can be visited (see Fig. 3). In order to represent moves of the order picker in the cross aisles, each pair of vertices $([i, f], [i+1, f])$ and $([i, b], [i+1, b])$ $(i = 1, \ldots, m-1$, where m denotes the number of picking aisles) is connected by two arcs. The depot is positioned in front of the leftmost picking aisle and identical to vertex $[1, f]$ in the graph. An example of a graph related to a (single-) block layout with five picking aisles is depicted in Fig. 4.

The weight c_a for an arc a in a picking aisle i can be determined as follows: The arc weight $c_{([i,2],[i,3])}$ is always equal to the largest gap between two adjacent pick locations in picking aisle i or a pick location and the adjacent cross aisle. Let j_i^* be the location represented by vertex $[i, 2]$. Then, three different cases have to be distinguished for the determination of $c_{([i,f],[i,1])}$ and $c_{([i,1],[i,2])}$:

(1) If j_i^* does not correspond to the location of a requested item but to the point where picking aisle i can be entered via the front cross aisle, then $c_{([i,f],[i,1])} = c_{([i,1],[i,2])} = 0$.
(2) If j_i^* corresponds to the pick location nearest to the front cross aisle, then $c_{([i,f],[i,1])}$ is equal to the distance between the front cross aisle and j_i^*, and $c_{([i,1],[i,2])} = 0$ holds.
(3) Otherwise, $c_{([i,f],[i,1])}$ is determined as in the second case, and $c_{([i,1],[i,2])}$ is the distance between the pick location nearest to the front cross aisle and j_i^*.

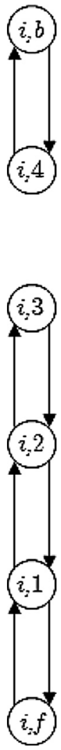


Fig. 5. Prohibited path in a solution.

The determination of $c_{([i,\,3],\,[i,\,4])}$ and $c_{([i,\,4],\,[i,\,5])}$ is performed analogously. If picking aisle *i* does not contain any requested item, then $c_{([i,\,2],\,[i,\,3])}$ is equal to the distance between the front and the rear cross aisle; the remaining arc weights are set to zero.

A feasible solution for the SPRP would be given by a tour which starts and ends at vertex [1, *f*] and includes the vertices [*i*, 1], [*i*, 2], [*i*, 3] and [*i*, 4] for each picking aisle *i* that contains at least one requested item. When applying a model formulation for the TSP to this graph, we therefore need degree constraints for these vertices in order to ensure that they are contained in the tour. However, this approach would result in two problems. First, even in an optimal solution for the SPRP, it is possible that vertices are visited more than once, which is not permitted in standard definitions of the TSP. Second, simply guaranteeing that all vertices in a picking aisle are visited does not necessarily ensure that a feasible solution for the SPRP is provided.

In Fig. 5, an infeasible combination of arcs in a picking aisle is depicted. All vertices in this picking aisle are visited, however, it cannot be guaranteed that all pick locations are included in the tour. This is due to the fact that the number of vertices, by which each picking aisle is defined, is not dependent on the number of requested items. Because of this reason, it is possible that some requested items are situated between the locations that are represented by the vertices. Vertices [*i*, 2] and [*i*, 3] represent the two locations defining the largest gap and, therefore, no pick location can be situated between those locations. This is not true for vertices [*i*, 1] and [*i*, 2] as well as for [*i*, 3] and [*i*, 4]. Since vertices [*i*, 1] and [*i*, 4] represent the location nearest and farthest from the front cross aisle, several storage locations containing requested items may be situated between these locations and the locations defining the largest gap. Thus, tours may be generated in which some of the pick locations are skipped.

In order to ensure feasibility of solutions for the SPRP which are obtained from a TSP formulation based on this graph, additional predecessor and successor constraints for arcs are needed. For example, we will have to ensure that arc ([*i*, 3], [*i*, 4]) is used if arc ([*i*, 2], [*i*, 3]) was chosen.

Another issue refers to the elimination of subtours. The general concept of the subtour elimination constraints in the TSP formulations presented above consists of the enumeration of the vertices according to the sequence in which they appear in the tour. This approach cannot be successful when using this graph, because on the one hand, vertices are allowed to be visited more than once and, on the other hand, some cycles are allowed within the tour (e.g. if a largest gap strategy is used in an aisle).

In order to avoid vertices to be visited more than once, vertices are split up into several vertices in such a way that each generated vertex can only be visited one time. According to Ratliff and Rosenthal (1983) vertices corresponding to cross aisles can be visited up to three times, while the other vertices may be visited at most twice. Therefore, we replace each vertex [*i*, *f*] and [*i*, *b*] $(i = 1, \ldots, m)$ by three vertices, where one vertex has to be used to enter a picking aisle and the other two vertices correspond to movements to the left and to the right in the cross aisles. The vertices [*i*, 1], [*i*, 2], [*i*, 3] and [*i*, 4] represent movements within a picking aisle *i* and are replaced by two vertices, where these vertices correspond to movements towards the rear cross aisle (up) and the front cross aisle (down), respectively. Furthermore, a vertex symbolizing the location of the depot is added. An example for the resulting graph is depicted in Fig. 6.

In general, the vertices of this graph can be described as follows. Vertex 0 symbolizes the location of the depot. The other vertices are characterized by a triple, where the first component represents the direction in which the tour can be continued. *r* and *l* indicate movements to the right and to the left, respectively. Movements towards the rear cross aisle and towards the front cross aisle are symbolized by *u* ("up") and *d* ("down"). The second component characterizes the number of picking aisle *i*, where picking aisle 1 is the leftmost and aisle *m* the rightmost picking aisle. The last component of the triple represents the location of the vertex, where *f* and *b* mean that the vertex corresponds to the front and the rear cross aisle, respectively. The four locations in a particular picking aisle are enumerated from 1 to 4. Based on this denotation, the vertices [*l*, 1, *b*], [*r*, *m*, *b*] and [*r*, *m*, *f*] do not exist, because at these points either moves to the left or to the right are possible. After having introduced the vertices of the graph, arcs are added based on the feasible options according to which requested items can be picked in a picking aisle (see Fig. 3).

These modifications result in a graph in which an optimal order picking tour can be constructed without visiting a vertex more than once. This graph includes more than twice the number of vertices as the previously presented graph does. However, the size of the improved model formulation for the SPRP will only be dependent on the number of arcs and not on the number of vertices. Furthermore, the size of the graph is completely independent of the number of pick locations. Applying a TSP formulation to this modified graph, in which the number of subtour elimination constraints increases linearly to the number of arcs, will lead to a mathematical model whose size increases linearly to the number of picking aisles.

4.2. Model formulation based on the modified graph

In this section, a TSP formulation is applied to the graph constructed in Section 4.1. The complete mathematical model for the SPRP is presented in the Appendix. It includes the following classes of constraints:

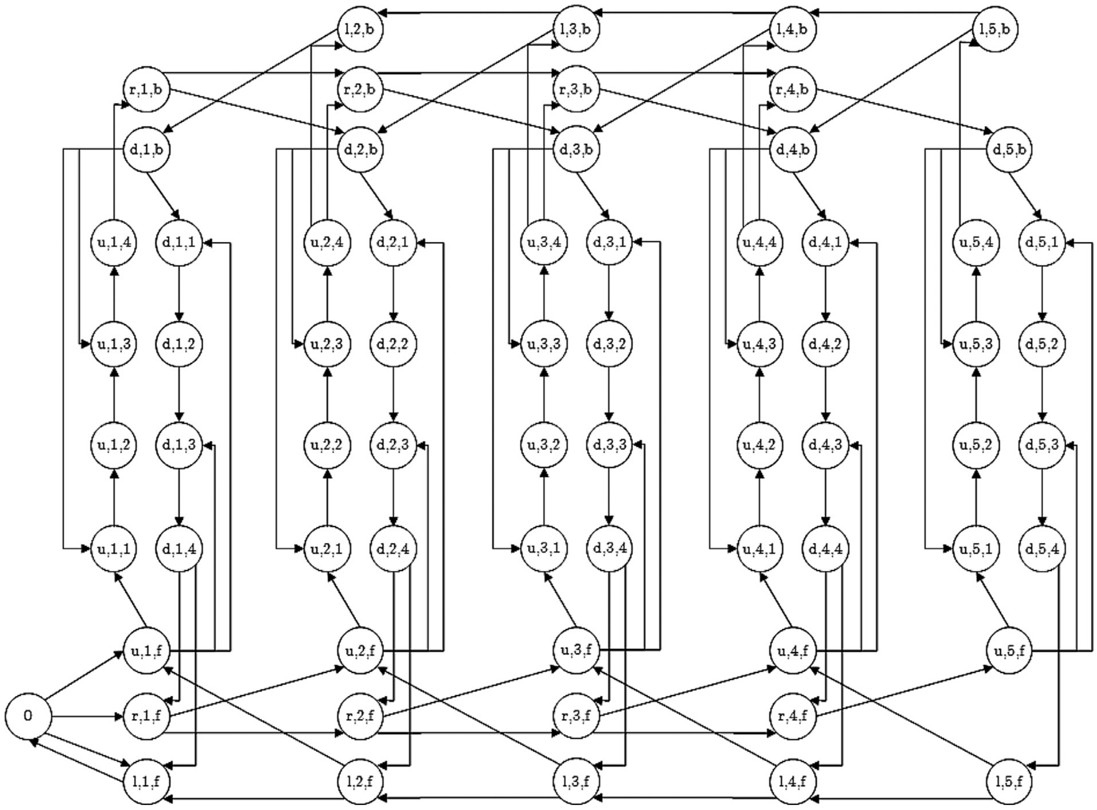


Fig. 6. Modified graph for a SPRP in a single-block layout with five picking aisles.

- Degree constraints [(37)–(59)]: Each vertex has to be left after it has been visited.
- Subtour elimination constraints [(60)–(90)]: The resulting tour has to be connected.
- Depot inclusion constraint [(91)]: The depot has to be included in the tour.
- Pick location inclusion constraints [(92)–(93)]: Each pick location has to be included in the tour.

The degree and subtour elimination constraints [(37)–(90)] are very similar to the corresponding constraints [(2)–(4), (8)–(11) and (15)–(20)] used in the general TSP formulations and will only be described briefly here. As decision variables we introduce binary variables for each arc, indicating if the arc is contained in the tour (variable is equal to 1) or not (0). For a picking aisle i, the notation of the variables for the different arc types is depicted in Fig. 7. For the sake of clarity, each arc type is only included for a single direction in this figure. Arcs corresponding to movements in the opposite direction are excluded. (For example, arc ([u, i, 3], [u, i, 4]) is depicted and ([d, i, 1], [d, i, 2]) is excluded.)

The degree constraints in general TSP formulations ensure that each vertex is visited exactly once, which means that the indegree and the outdegree of each vertex are equal to one, respectively. When dealing with a SPRP represented by the modified graph, each vertex is visited at most once. However, some vertices may exist which are not included in an optimal tour. Therefore, it has to be ensured that the indegree and the outdegree are equal to 1 if a vertex is visited and equal to 0 otherwise. This can be done by requiring that, for each vertex, the indegree is equal to the outdegree. In the degree constraints (37)–(59), the outdegree is calculated on the left hand side of the equation while the right hand side represents the indegree.

The mathematical models for the general TSP presented in Section 3 only differ in the way how subtours are excluded. Based on pretests, we decided to apply the subtour elimination constraints by Gavish and Graves (1978) in which the arcs are enumerated according to the sequence in which they are used in the tour. Constraints (10) ensure for each vertex $i \in V \backslash \{0\}$ that the sum of variables corresponding to arcs, which can be used to reach vertex i, has to be by one greater than the sum of variables for arcs leaving this vertex. Constraints (11) result in a solution where both sums contain exactly one variable greater than zero, respectively. The general principle of constraints (10) is also used in the model formulation based on the modified graph. The application of these constraints leads to constraints (60)–(80). The structure of the left hand side of these equations is equal to those of constraints (10). However, the right hand side of constraints (60)–(80) cannot be equal to one for each vertex, because this would lead to tours in which all vertices have to be visited although it is allowed to skip some vertices when using the modified graph. Therefore, the right hand side of constraints (60)–(80) has to be equal to zero if the corresponding vertex is not included in the tour; otherwise, it has to be equal to 1. This can be obtained by calculating the degree (here: outdegree) for each vertex. The second part of the subtour elimination constraints [(81)–(90)] is equivalent to (11).

Constraint (91) ensures that the depot is included in the tour.

Since not all vertices must necessarily be visited, we will have to introduce vertex-related criteria which have to be satisfied if it is permitted to skip a vertex. Each vertex representing an intersection between a picking and a cross aisle is allowed to be skipped.

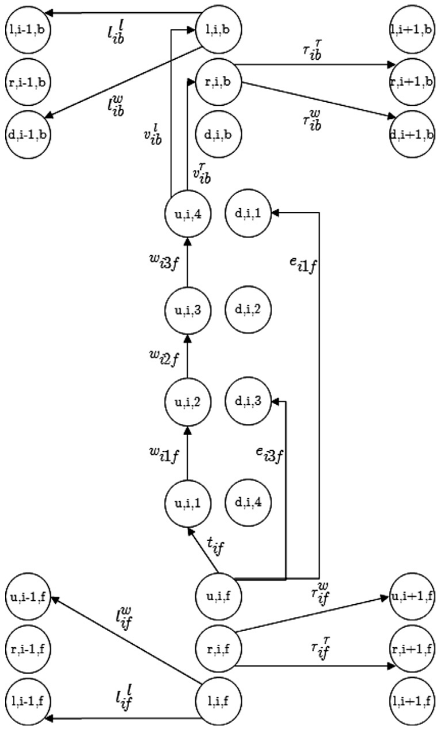


Fig. 7. Notation of the variables in the model formulation.

Furthermore, in some cases it is also possible to skip vertices corresponding to a pick location. The pick locations of a picking aisle i are represented by four pairs of vertices. In order to guarantee that all requested items are contained in the tour, it is sufficient to visit the vertices nearest to the cross aisles. In this case, the degree constraints ensure that all pick locations will be visited. Let us consider the two vertices nearest to the front cross aisle which are denoted by $[u, i, 1]$ and $[d, i, 4]$. Both vertices represent the same pick location and, therefore, only one of them has to be included in the tour. If vertex $[u, i, 1]$ is visited, the next vertex to be visited is $[u, i, 2]$, which means that the variable w_{i1f} has to be equal to one. If $[d, i, 4]$ is included in the tour, the vertex $[d, i, 3]$ has to be visited before, which implies $w_{i3b} = 1$. Since picking aisles which do not contain any requested items can also be skipped, the constraint $w_{i1f} + w_{i3b} \geq 1$ must hold for all picking aisles i containing at least one requested item. This is expressed by the constant b_i which is equal to 1 if picking aisle i has to be visited and 0 otherwise. Analogously, the constraints resulting from the pair of vertices nearest to the rear cross aisle can be constructed. This results in the following two constraints for a picking aisle i:

$$w_{i1f} + w_{i3b} \geq b_i \tag{32}$$

$$w_{i1b} + w_{i3f} \geq b_i \tag{33}$$

However, another special case exists in which one of these two pairs of vertices is allowed not to be contained in the tour. This case occurs if the largest gap is not defined by two pick locations but by a pick location and the adjacent cross aisle. If the corresponding cross aisle is the rear cross aisle, then the pair of vertices nearest to the rear cross aisle does not have to be visited. In this case, the vertices $[d, i, b]$, $[u, i, 4]$ and $[u, i, 3]$ represent the same location and, therefore, the distance between $[d, i, b]$ and $[u, i, 3]$, denoted by c^{e}_{i3b}, is equal to zero. This implies that constraint (33) must hold if and only if $c^{e}_{i3b} > 0$ which can be obtained by multiplying both sides of the constraint by c^{e}_{i3b} resulting in (93). The same line of argumentation holds if the front cross aisle is considered. In this case, multiplying constraint (32) by c^{e}_{i3f} leads to (92).

In total, we have $O(m)$ variables and constraints, i.e. the number of variables and constraints increases only linearly to the number of picking aisles m and is not dependent on the number of pick locations.

5. Numerical experiments

5.1. Design

In order to evaluate the above-presented formulations, we have performed numerical experiments for various classes of problem instances. The formulations will be compared w.r.t. the actual model sizes and the computing times needed by a commercial IP-solver for providing optimal solutions to these instances. For the numerical experiments, the characteristics of the warehouse have been chosen according to the experiments of Henn and Wäscher (2012). Each picking aisle consists of 90 storage locations, 45 on each side. Demands are uniformly distributed, i.e. each pick location has the same probability of being included in a picking order. The length of each storage location amounts to one length unit (LU). Whenever leaving an aisle, the order picker has to move one LU in the vertical direction from either the first or the last storage location in order to reach the cross aisle. The distance between two adjacent picking aisles is equal to 5 LU. The depot is assumed to be located in front of the leftmost picking aisle.

In their experiments, Henn and Wäscher (2012) fixed the number of picking aisles m to 10. Since both the size of the Steiner TSP formulation and the newly-proposed formulation are dependent on the number of picking aisles, we consider different values of m here. Therefore, m has been set to 5, 10, 15, 20, 25 and 30. For the size of the picking order (number of pick locations n), the values 30, 45, 60, 75 and 90 have been chosen. Combination of these parameters gives rise to 30 problem classes. For each class, 30 instances have been generated, resulting in 900 instances in total.

All formulations have been implemented and solved by CPLEX 12.6. The experiments have been carried out on a desktop PC with a 3.4 gigahertz Pentium processor with 8 gigabytes RAM. The computing time for each instance and formulation has been limited to 30 minutes.

5.2. Results

Before the model formulations are evaluated with respect to the corresponding computing times and the number of optimal solutions obtained within the time limit, they are compared regarding their sizes in dependency on the number of pick locations n and picking aisles m. For each problem class, the size of each formulation is depicted in Table 1.

Concerning the formulations of Miller et al. (1960) [MTZ], Gavish and Graves (1978) [GG] and Claus (1984) [C], the number of variables (#var) and constraints (#cons) depends on the number of pick locations n, whereas the size of the Steiner TSP formulation provided by Letchford et al. (2013) [LNT] is dependent on both the number of pick locations n and the number of picking aisles m. In comparison to this, the size of the newly-proposed formulation [SHSW] is determined by the number of picking aisles m only. In fact, the size of the new model formulation increases only linearly to the number of picking aisles. Therefore, it can be assumed that the new formulation for the SPRP outperforms the other formulations (w.r.t. the size of the models) if the ratio n/m gets large. This is actually confirmed by the data presented in Table 1. Only for a large number of picking aisles ($m = 30$) and a small number of

Table 1
Size of mathematical programming formulations for the SPRP.

(m, n)	MTZ		GG		C		LNT		SHSW	
	#var	#cons	#var	#cons	#var	#cons	#var	#cons	#var	#cons
(5, 30)	960	932	1830	992	28,830	28,892	583	1022	220	254
(5, 45)	2115	2072	4095	2162	95,220	95,312	748	1352	220	254
(5, 60)	3720	3662	7260	3782	223,260	223,382	913	1682	220	254
(5, 75)	5775	5702	11,325	5852	433,200	433,352	1078	2012	220	254
(5, 90)	8280	8192	16,290	8372	745,290	745,472	1243	2342	220	254
(10, 30)	960	932	1830	992	28,830	28,892	1638	2682	460	524
(10, 45)	2115	2072	4095	2162	95,220	95,312	1953	3312	460	524
(10, 60)	3720	3662	7260	3782	223,260	223,382	2268	3942	460	524
(10, 75)	5775	5702	11,325	5852	433,200	433,352	2583	4572	460	524
(10, 90)	8280	8192	16,290	8372	745,290	745,472	2898	5202	460	524
(15, 30)	960	932	1830	992	28,830	28,892	3193	5042	700	794
(15, 45)	2115	2072	4095	2162	95,220	95,312	3658	5972	700	794
(15, 60)	3720	3662	7260	3782	223,260	223,382	4123	6902	700	794
(15, 75)	5775	5702	11,325	5852	433,200	433,352	4588	7832	700	794
(15, 90)	8280	8192	16,290	8372	745,290	745,472	5053	8762	700	794
(20, 30)	960	932	1830	992	28,830	28,892	5248	8102	940	1064
(20, 45)	2115	2072	4095	2162	95,220	95,312	5863	9332	940	1064
(20, 60)	3720	3662	7260	3782	223,260	223,382	6478	10,562	940	1064
(20, 75)	5775	5702	11,325	5852	433,200	433,352	7093	11,792	940	1064
(20, 90)	8280	8192	16,290	8372	745,290	745,472	7708	13,022	940	1064
(25, 30)	960	932	1830	992	28,830	28,892	7803	11,862	1180	1334
(25, 45)	2115	2072	4095	2162	95,220	95,312	8568	13,392	1180	1334
(25, 60)	3720	3662	7260	3782	223,260	223,382	9333	14,922	1180	1334
(25, 75)	5775	5702	11,325	5852	433,200	433,352	10,098	16,452	1180	1334
(25, 90)	8280	8192	16,290	8372	745,290	745,472	10,863	17,982	1180	1334
(30, 30)	960	932	1830	992	28,830	28,892	10,858	16,322	1420	1604
(30, 45)	2115	2072	4095	2162	95,220	95,312	11,773	18,152	1420	1604
(30, 60)	3720	3662	7260	3782	223,260	223,382	12,688	19,982	1420	1604
(30, 75)	5775	5702	11,325	5852	433,200	433,352	13,603	21,812	1420	1604
(30, 90)	8280	8192	16,290	8372	745,290	745,472	14,518	23,642	1420	1604
	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^3)$	$O(n^3)$	$O(n^2m)$	$O(n^2m)$	$O(m)$	$O(m)$

pick locations ($n = 30$) – resulting in a very small ratio n/m – the number of variables and constraints in the SHSW formulation may turn out to be larger than in the MTZ and GG formulations, even though the difference does not appear to be really significant. The C and the LNT formulations are clearly outperformed by the SHSW formulation.

It can be expected that the performance of a formulation is strongly correlated with its size. In this case, a good performance means that by applying the model formulation an optimal solution can be found within a small amount of computing time. For each formulation, Table 2 depicts the number of optimal solutions obtained within 30 minutes.

The application of the general TSP formulations MTZ, GG and C leads to rather unsatisfactory results. As expected, the number of optimal solutions obtained within the predefined time limit significantly decreases with an increasing number of pick locations n. Furthermore, it should be noted that the performance of these three formulations is also slightly dependent on the number of picking aisles m. The impact of m is not as strong as it is the case for n, but with an increasing number of picking aisles the number of instances in which an optimal solution can be found within 30 minutes of computing time also decreases. When comparing the results from the three general TSP formulations, it can be seen that the MTZ formulation shows the worst performance. Although requiring the least number of variables and constraints, no problem class exists in which all instances can be solved to optimality. This performance can be explained by the fact that solving its LP relaxation leads to a very weak lower bound. By application of the C formulation all problem instances with a small number of pick locations ($n = 30$) can be solved to optimality within 30 minutes. However, when increasing n, the performance gets much worse. For instances with 60 or more pick locations only a single

Table 2
Number of solved instances (out of 30) within 30 minutes of computing time.

(m, n)	MTZ	GG	C	LNT	SHSW
(5, 30)	28	**30**	**30**	**30**	**30**
(5, 45)	19	**30**	15	**30**	**30**
(5, 60)	3	28	0	**30**	**30**
(5, 75)	0	22	0	**30**	**30**
(5, 90)	0	8	0	**30**	**30**
(10, 30)	26	**30**	**30**	**30**	**30**
(10, 45)	19	**30**	13	**30**	**30**
(10, 60)	1	**30**	0	**30**	**30**
(10, 75)	0	25	0	**30**	**30**
(10, 90)	0	8	0	**30**	**30**
(15, 30)	26	**30**	**30**	**30**	**30**
(15, 45)	6	**30**	7	**30**	**30**
(15, 60)	0	29	1	**30**	**30**
(15, 75)	0	21	0	**30**	**30**
(15, 90)	0	9	0	20	**30**
(20, 30)	24	**30**	**30**	**30**	**30**
(20, 45)	5	**30**	5	**30**	**30**
(20, 60)	0	26	0	**30**	**30**
(20, 75)	0	11	0	23	**30**
(20, 90)	0	4	0	7	**30**
(25, 30)	18	**30**	**30**	**30**	**30**
(25, 45)	0	**30**	2	**30**	**30**
(25, 60)	0	26	0	**30**	**30**
(25, 75)	0	14	0	19	**29**
(25, 90)	0	3	0	2	**24**
(30, 30)	16	**30**	**30**	**30**	**30**
(30, 45)	0	**30**	3	**30**	**30**
(30, 60)	0	29	0	**30**	**30**
(30, 75)	0	10	0	16	**24**
(30, 90)	0	2	0	0	**21**

problem instance can be solved. Therefore, this formulation can only be used for very small instances which could be expected since this formulation requires the largest number of variables and constraints with a size increasing cubically to the number of pick locations. The GG formulation is the best TSP formulation in our numerical experiments. Using this formulation, we can obtain optimal solutions for all instances with up to 45 pick locations and even for $n = 60$, optimal solutions can be found within the given time limit for most instances. However, since the number of variables and constraints increases with an increasing number of pick locations (as it is also the case for the other TSP formulations), only a small fraction of the problem instances can be solved to optimality when n gets quite large ($n = 90$).

More convincing results are obtained when the LNT and the SHSW formulations are applied. For all instances with up to 10 picking aisles or 60 pick locations, an optimal solution can be found within 30 minutes of computing time. The size of both formulations is dependent on the number of picking aisles and this can also be seen in the results since the number of optimal solutions obtained within the time interval decreases with an increasing number of picking aisles. By using the LNT formulation all instances with up to 10 picking aisles can be solved whereas the SHSW formulation provides optimal solution for all instances with up to 20 picking aisles. An increasing number of pick locations also shows a negative impact on the performance of both formulations. While the number of optimal solutions obtained by applying the SHSW formulation only slightly decreases with an increasing number of pick locations, the performance of the LNT formulation drastically deteriorates. For $m \leq 15$, the LNT formulation leads to quite convincing results. When considering a larger number of picking aisles m, it can be seen that the LNT formulation still provides optimal solutions for all instances with up to 60 pick locations. However, for instances with $m \geq 20$ only a few optimal solutions can be found by using this formulation when the number of pick locations is very large ($n = 90$). Especially, when looking at the results from the largest problem class ($m = 30, n = 90$), we can see the limitation of the Steiner TSP formulation. While the SHSW formulation still provides optimal solutions for 21 out of 30 (70%) of those problem instances, no optimal solution can be found by using the LNT formulation. The superiority of the SHSW formulation for this case can be attributed to the size of the two formulations. Since the number of variables and constraints required by the LNT formulation is not only dependent on the number of picking aisles m but also on the number of pick locations n, an increase of both m and n has a strong impact on the performance of the LNT formulation.

In Table 3 the average computing times for the five formulations are presented. Computing times have only been recorded if the instance has been solved to optimality and, therefore, no information about computing times is given for some problem classes when the MTZ, the C or the LNT formulation has been applied.

The results depicted in Table 3 are in accordance to the expectations raised by comparing the sizes of the formulations and the number of optimal solutions obtained within the predefined time interval. Again, the general TSP formulations lead to the worst results. When applying one of these formulations, the computing times grow very fast if the number of pick locations is increased. The strongest increase of computing time can be observed for the C formulation since its size increases cubically to the number of picking aisles (instead of quadratically as it is the case for the MTZ and GG formulation). For the MTZ formulation, the computing times are already quite large for a small number of pick locations ($n = 30$), while computing times for other problem classes cannot be interpreted since only a few instances can be optimally solved and, therefore, computing times have not been recorded for most instances. As expected, the GG formulation requires the lowest computing times. For the solution of instances with 30 pick locations, on average less than 5 seconds are required and even for instances with $n = 45$, the application of the GG formulation results in an optimal solution within approximately one minute or less. However, with a further increase of n, we can observe a very strong increase of the computing times, which leads us to the conclusion that this formulation should not be applied to instances with 60 or more pick locations. Furthermore, not only an increasing number of pick locations n but also an increase in the number of picking aisles m seem to have a negative impact on the computing times required by general TSP formulations, even though the size of these formulations is not dependent on m.

Table 3
Computing times [seconds].

(m, n)	MTZ	GG	C	LNT	SHSW
(5, 30)	109.18	2.65	25.67	2.84	**0.09**
(5, 45)	869.72	22.37	1169.04	8.71	**0.09**
(5, 60)	1666.20	453.94	–	25.66	**0.09**
(5, 75)	–	898.21	–	63.22	**0.09**
(5, 90)	–	1393.63	–	146.31	**0.10**
(10, 30)	310.62	1.94	114.14	4.57	**1.60**
(10, 45)	991.73	14.59	1408.99	14.66	**1.03**
(10, 60)	1750.94	90.74	–	37.09	**1.42**
(10, 75)	–	482.53	–	156.22	**1.36**
(10, 90)	–	1414.21	–	303.68	**0.62**
(15, 30)	372.77	3.40	89.14	7.45	**2.29**
(15, 45)	1564.07	20.20	1562.30	24.85	**5.28**
(15, 60)	–	395.01	1761.62	90.30	**10.64**
(15, 75)	–	1069.54	–	357.27	**15.10**
(15, 90)	–	1537.30	–	811.61	**19.41**
(20, 30)	555.07	**4.05**	104.96	9.47	10.57
(20, 45)	1649.94	36.64	1656.18	41.30	**27.32**
(20, 60)	–	524.44	–	147.52	**114.33**
(20, 75)	–	1551.92	–	614.11	**216.63**
(20, 90)	–	1780.10	–	1627.68	**485.71**
(25, 30)	899.22	**4.30**	110.76	15.07	54.46
(25, 45)	–	62.09	1768.15	**41.55**	85.46
(25, 60)	–	497.17	–	**173.87**	258.92
(25, 75)	–	1256.74	–	858.44	**527.39**
(25, 90)	–	1758.36	–	1764.21	**646.59**
(30, 30)	1098.95	**4.18**	98.49	14.00	204.18
(30, 45)	–	59.64	1760.77	**43.01**	406.19
(30, 60)	–	418.95	–	**293.87**	508.80
(30, 75)	–	1510.33	–	1102.47	**638.89**
(30, 90)	–	1762.80	–	–	**786.29**

Apart from very small instances with only 30 pick locations, the LNT and SHSW formulations require the smallest amount of computing time. For both formulations, the computing times increase with both an increasing number of pick locations and picking aisles. For the LNT formulation, this could be expected since the size of the mathematical model is also dependent on both numbers and, therefore, it is not surprising that an increasing number of pick locations has a much larger impact on the LNT than on the SHSW formulation. However, the results show that computing times required for the solution of the SHSW formulation also increase with an increasing number of pick locations, even though the number of arcs and vertices of the modified graph is independent of this number. This is due to two reasons: First, constraints (92) and (93) are redundant for a picking aisle i if $b_i = 0$, i.e. if picking aisle i does not contain any pick locations. The less pick locations have to be considered, the smaller the probability gets that a large number of picking aisles has to be visited. Second, a large number of pick locations results in many solutions almost as good as an optimal solution and, therefore, proving optimality gets quite time consuming.

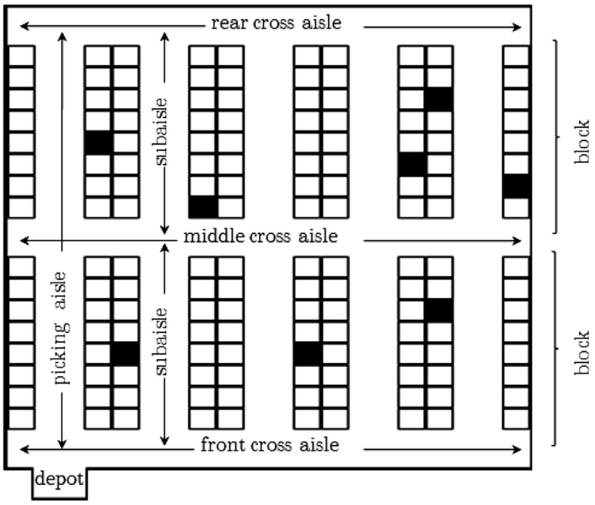


Fig. 8. Two-block layout.

As could be expected with respect to the number of constraints and variables, the application of the SHSW formulation outperforms the other approaches by far when the number of picking aisles is not too large ($m \leq 15$). If the number of aisles is small ($m = 5$), by using the SHSW formulation the computing times can be reduced by a factor between 30 ($n = 30$) and 1500 ($n = 90$) compared to the best of the other four formulations. As mentioned before, in term of the size, the advantage of the SHSW formulation over general and Steiner TSP formulations diminishes if the ratio n/m gets small. This observation coincides with the computing times depicted in Table 3. For instances with a larger number of picking aisles ($m \geq 20$) and a small number of pick locations ($n = 30$), the GG formulation requires substantial less computing time than the SHSW formulation. If the number of picking aisles is very large ($m \geq 25$) and only up to 60 locations have to be visited, the LNT formulation clearly outperforms the SHSW formulation. However, since the size of the general TSP and the LNT formulations is dependent on the number of pick locations n, the performance of these approaches deteriorates when n gets large, whereas the SHSW formulation provides optimal solutions within a decent amount of computing time even for large instances.

6. Some extensions to the proposed model formulation

In the previous sections, we focused on warehouses with a single-block layout and developed a graph representing the SPRP in such warehouses and a corresponding mathematical programming formulation. In the following, we will present several modifications which can be used for representing variants of the SPRP which are highly relevant to practice.

6.1. SPRP in multi-block layouts

Single-block layouts result in long picking aisles if the number of picking aisles is small and the number of storage locations is large. Consequently, the order pickers will have to travel long distances to change over from one picking aisle to another. This becomes particularly inefficient if only a few items have to be removed from each picking aisle. Therefore, picking aisles are usually arranged in several blocks which are separated by additional cross aisles. In Fig. 8, a warehouse with two blocks and one additional (middle) cross aisle is depicted. The middle cross aisle divides each picking aisle into two subaisles.

Roodbergen and de Koster (2001a) have extended the algorithm of Ratliff and Rosenthal (1983) for the SPRP in warehouses with two blocks. However, with an increasing number of cross aisles, the SPRP becomes harder to solve since the number of options for changing over from one picking aisle to another increases. Instead of 7 different equivalence classes (in the case for the original algorithm of Ratliff & Rosenthal, 1983) now 25 equivalence classes have to be dealt with in each phase of a dynamic programming procedure. An extension to warehouses with three or more blocks has not been proposed in the literature so far but can be considered even more complex.

The graph and the corresponding mathematical programming formulation presented in this paper can be extended easily to warehouse layouts with an arbitrary number of blocks. For this purpose, a slightly modified copy of the graph has to be added for each additional block. The modification refers to the vertices corresponding to points in the middle cross aisles. In a single-block layout, at each point in a cross aisle, the picker can choose between moving to the left, moving to the right or entering the respective picking aisle. Three different vertices are needed to represent these movement options. In a two-block or a multi-block layout, when positioned in a middle cross aisle, now the order picker has an additional option, namely to choose between entering the picking aisle in the upper block and entering the picking aisles in the lower block. Therefore, each point of a middle cross aisle has to be represented by four vertices.

6.2. SPRP with decentralized depositing

In the standard SPRP, a central depot is assumed, i.e. the picker starts and terminates the tour at the depot. In practice, items can often be deposited at the end (head) of each picking aisle (de Koster & van der Poort, 1998). In other words, the starting point of a route is known in advance, which can now be the depot or any picking-aisle head. The point where the picker terminates the tour has to be determined. The algorithm of Ratliff and Rosenthal (1983) has been modified by de Koster and van der Poort (1998) for this case of decentralized depositing. However, this modification is only applicable to picking areas following a single-block layout.

Again, by simple modifications to the graph and the mathematical model presented above, this case and also its extension to a multi-block layout can be dealt with in the approach proposed in this paper: The (given) starting point will be represented by vertex 0. Since the tour may terminate at the head of any picking aisle, a vertex D is added for each picking aisle representing a location where items can be deposited. Furthermore, for each picking aisle i, an arc is added, connecting vertex D and vertex $[d, i, 4]$, the latter corresponding to the pick location in this picking aisle closest to the front cross aisle. An example of a graph for decentralized depositing with five picking aisles is depicted in Fig. 9. In the resulting model formulation, we only have to ensure that the outdegree of vertex 0 and the sum of the indegrees of the vertices D and vertex 0 are equal to 1.

6.3. Generating simple routes for the SPRP

Order pickers in practice seem to prefer simple tours through the warehouse which can be memorized easily. More complex tours, e.g. those resulting from the application of exact algorithms, may contain elements which are looked upon as unnatural, resulting in an additional orientation effort, which reduces the travel velocity of the pickers, increases the risk of making a false step and lengthening the tour, and also increases the number of picking errors (Petersen & Schmenner, 1999). In the worst case, the proposed tours may not be accepted at all, and the pickers will compile

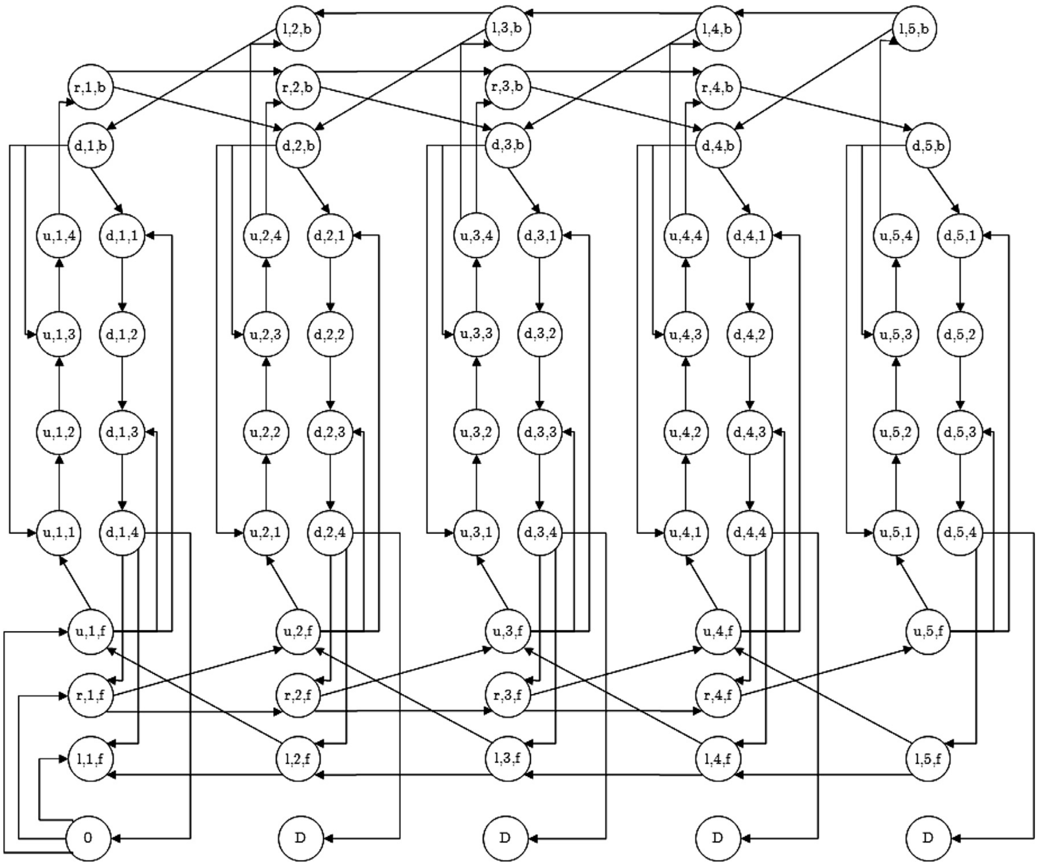


Fig. 9. Graph for a SPRP with decentralized depositing.

paths of their own through the warehouse which they may consider good but may in fact be far from optimal.

In order to generate more acceptable, simple tours, one can try to eliminate elements from the tours which may be considered as unnatural. One such element refers to the fact that a picking aisle is entered more than once (Roodbergen & de Koster, 2001b). Prohibiting a picking aisle to be entered more than once is not straightforward in the TSP formulations for the SPRP, but it can easily be integrated in the graph and the corresponding model proposed here. A solution in which each subaisle is entered at most once corresponds to the application of either the return or the traversal strategy, while the largest-gap strategy must be excluded. This can be achieved by deleting the edges e_{i3f} and e_{i3b} from the graph for each subaisle *i*. Solving the resulting model formulation will result in a simple tour not longer than the tour provided by the combined heuristic of Roodbergen and de Koster (2001b).

The simplicity of a picking tour is also seen related to the number of direction changes the order picker has to perform within the cross aisles. First, tours with frequent changes of the moving direction happen to be quite confusing for the order picker since the picking aisles are not visited in a natural sequence (Petersen & Schmenner, 1999). Second, movements within cross aisles only serve the purpose of changing from one picking aisle to another. No items are collected while moving in a cross aisle. If a tour includes frequent direction changes within the cross aisles, the order picker may conclude that the travel distance can be reduced by deviating from the tour plan, resulting in an even longer tour. Therefore, it can be useful to limit the number of direction changes or even prohibit them.

The proposed model formulation for the SPRP can easily be modified to include this feature. Let the order picker be in an arbitrary cross aisle. For a given pair of adjacent picking aisles $(i, i+1)$, it is possible to change over from picking aisle i to $i+1$ (movement to the right) and from aisle $i+1$ to i (movement to the left). In the mathematical model, these movements can be represented by two variables each. In order to prohibit a direction change in the corresponding cross aisle, the sum of these variables has to be smaller than or equal to 1 for each pair of adjacent picking aisles. For example, in order to prohibit a direction change in the front cross aisle, the following constraints have to be added:

$$l_{if}^{w} + l_{if}^{l} + r_{i-1,f}^{w} + r_{i-1,f}^{r} \leq 1 \qquad \forall\, i = 2, \ldots, m-1 \tag{34}$$

$$l_{mf}^{w} + l_{mf}^{l} + r_{m-1,f}^{w} \leq 1 \tag{35}$$

By introducing such constraints for a certain number of cross aisles, it is possible to limit the number of direction changes within cross aisles, resulting in tours which may appear to be more straightforward for the order picker and can be memorized more easily.

7. Conclusion and outlook

In this article, we considered the Single-Picker Routing Problem in a single-block layout which represents a special case of the Traveling Salesman Problem. We proposed the first mathematical programming formulation that adopts the specific properties of this problem. It could be shown that the size of the proposed formulation (expressed in the number of variables and constraints) depends on the number of picking aisles only and not on the number of storage locations to be visited.

By means of numerical experiments, the proposed formulation has been evaluated and compared to more general TSP and Steiner TSP formulations. It could be demonstrated that the proposed formulation is advantageous with respect to the size of the formulation. The proposed formulation also outperforms the general formulations with respect to computing times. Moreover, within a given time limit, we were able to find optimal solutions for instances which could not be solved by means of the general formulations. The numerical results clearly indicate that for specific, efficiently-solvable cases of the Traveling Salesman Problem, the development of customized formulations is necessary in order to obtain optimal solutions within short computing times.

This work focused on the SPRP in single-block layouts, but it has also been shown how several practically relevant situations (multi-block layout, decentral depositing, generation of simple tours) can be integrated into the proposed formulation. Interesting topics of future research could be dealing with other features encountered with order picking systems in practice. This may refer to the determination of picker tours in warehouses with high racks where movements within the picking aisles are much more complex.

Furthermore, for the proposed formulation wide-aisles have been assumed which enable order pickers to pass and overtake each other. As a consequence, their tours through the warehouse can be determined independently from each other. In narrow-aisle warehouses, a picking aisle may be blocked when a picker cannot enter the aisle before it is left by another picker moving in the opposite direction. Even traffic jams may occur when two pickers have to collect items from the same storage location at the same time. Picker tours can no longer be determined independently but will have to be planned simultaneously with the batching of customer orders into picking orders. An optimization model for order batching has been suggested by Hong, Johnson, and Peters (2012) which takes blocking considerations into account. However, this model is restricted to warehouses in which only one-way travel within picking aisles is allowed. Therefore, combining picker blocking aspects with more general routing schemes and their integration into the proposed model formulation could be another promising area of future research.

Appendix. Model formulation for the SPRP

Sets:

$I = \{1, \ldots, m\}$: set of picking aisles;
$F = \{f, b\}$: set of cross aisles.

Constants:

$$b_i = \begin{cases} 1, & \text{if picking aisle } i \in I \text{ contains at least one requested item,} \\ 0, & \text{otherwise;} \end{cases}$$

c^a : distance between two adjacent aisles;

c^0 : distance between the depot and the location on the front cross aisle where the first picking aisle can be entered;

$c^e_{is\alpha}$: distance between front cross aisle ($\alpha = f$) and vertex $[d, i, s]$ ($s \in \{1, 3\}$) or rear cross aisle ($\alpha = b$) and vertex $[u, i, s]$ in picking aisle $i \in I$;

$c^t_{i\alpha}$: distance between front cross aisle ($\alpha = f$) and vertex $[u, i, 1]$ or back cross aisle ($\alpha = b$) and vertex $[d, i, 1]$ in picking aisle $i \in I$;

c^w_{is} : distance between location $s \in \{1, 2, 3\}$ and location $s + 1$ in picking aisle $i \in I$;

M : large number (e.g. number of vertices).

Binary variables indicating the edges included in the tour:

$$r^r_{i\alpha} = \begin{cases} 1, & \text{if edge } ([r, i, \alpha], [r, i+1, \alpha]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad (i, \alpha) \in (I \setminus \{m-1, m\}) \times F;$$

$$r^w_{ib} = \begin{cases} 1, & \text{if edge } ([r, i, b], [d, i+1, b]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad i \in I \setminus \{m\};$$

$$r^w_{if} = \begin{cases} 1, & \text{if edge } ([r, i, f], [u, i+1, f]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad i \in I \setminus \{m\};$$

$$\ell^\ell_{i\alpha} = \begin{cases} 1, & \text{if edge } ([\ell, i, \alpha], [\ell, i-1, \alpha]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad (i, \alpha) \in ((I \setminus \{1, 2\}) \times F) \cup \{(2, f)\};$$

$$\ell^w_{ib} = \begin{cases} 1, & \text{if edge } ([\ell, i, b], [d, i-1, b]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad i \in I \setminus \{1\};$$

$$\ell^w_{if} = \begin{cases} 1, & \text{if edge } ([\ell, i, f], [u, i-1, f]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad i \in I \setminus \{1\};$$

$$e_{isb} = \begin{cases} 1, & \text{if edge } ([d, i, b], [u, i, s]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad (i, s) \in I \times \{1, 3\};$$

$$e_{isf} = \begin{cases} 1, & \text{if edge } ([u, i, f], [d, i, s]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad (i, s) \in I \times \{1, 3\};$$

$$t_{ib} = \begin{cases} 1, & \text{if edge } ([d, i, b], [d, i, 1]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad i \in I;$$

$$t_{if} = \begin{cases} 1, & \text{if edge } ([u, i, f], [u, i, 1]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad i \in I;$$

$$w_{isb} = \begin{cases} 1, & \text{if edge } ([d, i, s], [d, i, s+1]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad (i, s) \in I \times \{1, 2, 3\};$$

$$w_{isf} = \begin{cases} 1, & \text{if edge } ([u, i, s], [u, i, s+1]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad (i, s) \in I \times \{1, 2, 3\};$$

$$v^r_{ib} = \begin{cases} 1, & \text{if edge } ([u, i, 4], [r, i, b]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad i \in I \setminus \{m\};$$

$$v^r_{if} = \begin{cases} 1, & \text{if edge } ([d, i, 4], [r, i, f]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases} \quad i \in I \setminus \{m\};$$

$$v_{ib}^{\ell} = \begin{cases} 1, & \text{if edge } ([u,i,4],[\ell,i,b]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases}$$
$i \in I \setminus \{1\}$;

$$v_{if}^{\ell} = \begin{cases} 1, & \text{if edge } ([d,i,4],[\ell,i,f]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases}$$
$i \in I$;

$$y_{\alpha}^{0} = \begin{cases} 1, & \text{if edge } ([0],[\alpha,1,f]) \text{ is contained in the tour,} \\ 0, & \text{otherwise,} \end{cases}$$
$\alpha \in \{l, r, u\}$;

$$y_{0}^{\ell} = \begin{cases} 1, & \text{if edge } ([\ell,1,f],[0]) \text{ is contained in the tour,} \\ 0, & \text{otherwise.} \end{cases}$$

Real-valued variables to exclude subcycles:

$\tilde{r}_{i\alpha}^{r}$, $(i,\alpha) \in (I \setminus \{m-1, m\}) \times F$;
$\tilde{r}_{i\alpha}^{w}$, $(i,\alpha) \in (I \setminus \{m\}) \times F$;
$\tilde{\ell}_{i\alpha}^{\ell}$, $(i,\alpha) \in ((I \setminus \{1,2\}) \times F) \cup \{(2,f)\}$;
$\tilde{\ell}_{i\alpha}^{w}$, $(i,\alpha) \in (I \setminus \{1\}) \times F$;
$\tilde{e}_{is\alpha}$, $(i,s,\alpha) \in I \times \{1,3\} \times F$;
$\tilde{t}_{i\alpha}$, $(i,\alpha) \in I \times F$;
$\tilde{w}_{is\alpha}$, $(i,s,\alpha) \in I \times \{1,2,3\} \times F$;
$\tilde{v}_{i\alpha}^{r}$, $(i,\alpha) \in (I \setminus \{m\}) \times F$;
$\tilde{v}_{i\alpha}^{\ell}$, $(i,\alpha) \in ((I \setminus \{1\}) \times F) \cup \{(1,f)\}$;
$\tilde{y}_{\alpha}^{0}$, $\alpha \in \{l, r, u\}$.

Objective function:

$$\begin{aligned} \min \ & \sum_{i=1}^{m-2} \sum_{\alpha \in F} c^a \cdot \left(r_{i\alpha}^{r} + r_{i\alpha}^{w} \right) + c^a \cdot \sum_{\alpha \in F} r_{m-1,\alpha}^{w} \\ & + \sum_{i=3}^{m} \sum_{\alpha \in F} c^a \cdot \left(\ell_{i\alpha}^{\ell} + \ell_{i\alpha}^{w} \right) + c^a \cdot \ell_{2f}^{\ell} + c^a \cdot \sum_{\alpha \in F} \ell_{2\alpha}^{w} \\ & + \sum_{i=1}^{m} \sum_{s \in \{1,3\}} \sum_{\alpha \in F} c_{is\alpha}^{e} \cdot e_{is\alpha} + \sum_{i=1}^{m} \sum_{\alpha \in F} c_{i\alpha}^{t} \cdot t_{i\alpha} \\ & + \sum_{i=1}^{m} \sum_{s=1}^{3} \sum_{\alpha \in F} c_{is\alpha}^{w} \cdot w_{is\alpha} + \sum_{i=1}^{m-1} \sum_{\alpha \in F} c_{i\alpha}^{t} \cdot v_{i\alpha}^{r} \\ & + \sum_{i=2}^{m} \sum_{\alpha \in F} c_{i\alpha}^{t} \cdot v_{i\alpha}^{\ell} + c_{1f}^{t} \cdot v_{1f}^{\ell} \\ & + c^0 \cdot \left(y_l^0 + y_r^0 + y_u^0 + y_0^l \right) \end{aligned} \tag{36}$$

Degree constraints:

- Constraints corresponding to the depot

$$y_l^0 + y_r^0 + y_u^0 = y_0^l \tag{37}$$

- Constraints corresponding to vertices $[r, i, \alpha]$

$$r_{i\alpha}^{r} + r_{i\alpha}^{w} = v_{i\alpha}^{r} + r_{i-1,\alpha}^{r} \quad \forall (i,\alpha) \in (I \setminus \{1, m-1, m\}) \times F \tag{38}$$

$$r_{m-1,\alpha}^{w} = v_{m-1,\alpha}^{r} + r_{m-2,\alpha}^{r} \quad \forall \alpha \in F \tag{39}$$

$$r_{1f}^{r} + r_{1f}^{w} = v_{1f}^{r} + y_r^0 \tag{40}$$

$$r_{1b}^{r} + r_{1b}^{w} = v_{1b}^{r} \tag{41}$$

- Constraints corresponding to vertices $[\ell, i, \alpha]$

$$\ell_{i\alpha}^{\ell} + \ell_{i\alpha}^{w} = v_{i\alpha}^{\ell} + \ell_{i+1,\alpha}^{\ell} \quad \forall (i,\alpha) \in (I \setminus \{1,2,m\}) \times F \tag{42}$$

$$\ell_{m\alpha}^{\ell} + \ell_{m\alpha}^{w} = v_{m\alpha}^{\ell} \quad \forall \alpha \in F \tag{43}$$

$$\ell_{2f}^{\ell} + \ell_{2f}^{w} = v_{2f}^{\ell} + \ell_{3f}^{\ell} \tag{44}$$

$$\ell_{2b}^{w} = v_{2f}^{\ell} + \ell_{3f}^{\ell} \tag{45}$$

$$y_0^{\ell} = y_{\ell}^0 + v_{1f}^{\ell} + \ell_{2f}^{\ell} \tag{46}$$

- Constraints corresponding to vertices $[u, i, f]$ and $[d, i, b]$

$$t_{i\alpha} + e_{i1\alpha} + e_{i3\alpha} = r_{i-1,\alpha}^{w} + \ell_{i+1,\alpha}^{w} \quad \forall (i,\alpha) \in (I \setminus \{2, m\}) \times F \tag{47}$$

$$t_{m\alpha} + e_{m1\alpha} + e_{m3\alpha} = r_{m-1,\alpha}^{w} \quad \forall \alpha \in F \tag{48}$$

$$t_{1f} + e_{11f} + e_{13f} = y_u^0 + \ell_{2f}^{w} \tag{49}$$

$$t_{1b} + e_{11b} + e_{13b} = \ell_{2b}^{w} \tag{50}$$

- Constraints corresponding to vertices $[u, i, 4]$ and $[d, i, 4]$

$$v_{i\alpha}^{r} + v_{i\alpha}^{\ell} = w_{i3\alpha} \quad \forall (i,\alpha) \in (I \setminus \{2, m\}) \times F \tag{51}$$

$$v_{m\alpha}^{\ell} = w_{m3\alpha} \quad \forall \alpha \in F \tag{52}$$

$$v_{1f}^{r} + v_{1f}^{\ell} = w_{13f} \tag{53}$$

$$v_{1b}^{r} = w_{13b} \tag{54}$$

- Constraints corresponding to vertices $[u, i, s]$ and $[d, i, s]$

$$w_{i1f} = t_{if} + e_{i1b} \quad \forall i \in I \tag{55}$$

$$w_{i1b} = t_{ib} + e_{i1f} \quad \forall i \in I \tag{56}$$

$$w_{i2\alpha} = w_{i1\alpha} \quad \forall (i,\alpha) \in I \times F \tag{57}$$

$$w_{i3f} = w_{i2f} + e_{i3b} \quad \forall i \in I \tag{58}$$

$$w_{i3b} = w_{i2b} + e_{i3f} \quad \forall i \in I \tag{59}$$

Subtour elimination constraints:

- Constraints corresponding to vertices $[r, i, \alpha]$

$$\tilde{v}_{i\alpha}^{r} + \tilde{r}_{i-1,\alpha}^{r} - \left(\tilde{r}_{i\alpha}^{r} + \tilde{r}_{i\alpha}^{w} \right) = r_{i\alpha}^{r} + r_{i\alpha}^{w} \quad \forall (i,\alpha) \in (I \setminus \{1, m-1, m\}) \times F \tag{60}$$

$$\tilde{v}_{m-1,\alpha}^{r} + \tilde{r}_{m-2,\alpha}^{r} - \tilde{r}_{m-1,\alpha}^{w} = r_{m-1,\alpha}^{w} \quad \forall \alpha \in F \tag{61}$$

$$\tilde{v}_{1f}^{r} + \tilde{y}_r^0 - \left(\tilde{r}_{1f}^{r} + \tilde{r}_{1f}^{w} \right) = r_{1f}^{r} + r_{1f}^{w} \tag{62}$$

$$\tilde{v}_{1b}^{r} - \left(\tilde{r}_{1b}^{r} + \tilde{r}_{1b}^{w} \right) = r_{1b}^{r} + r_{1b}^{w} \tag{63}$$

- Constraints corresponding to vertices $[\ell, i, \alpha]$

$$\tilde{v}_{i\alpha}^{\ell} + \tilde{\ell}_{i+1,\alpha}^{\ell} - \left(\tilde{\ell}_{i\alpha}^{\ell} + \tilde{\ell}_{i\alpha}^{w} \right) = \ell_{i\alpha}^{\ell} + \ell_{i\alpha}^{w} \quad \forall (i,\alpha) \in (I \setminus \{1,2,m\}) \times F \tag{64}$$

$$\tilde{v}_{m\alpha}^{\ell} - \left(\tilde{\ell}_{m\alpha}^{\ell} + \tilde{\ell}_{m\alpha}^{w} \right) = \ell_{m\alpha}^{\ell} + \ell_{m\alpha}^{w} \quad \forall \alpha \in F \tag{65}$$

$$\tilde{v}_{2f}^{\ell} + \tilde{\ell}_{3f}^{\ell} - \left(\tilde{\ell}_{2f}^{\ell} + \tilde{\ell}_{2f}^{w} \right) = \ell_{2f}^{\ell} + \ell_{2f}^{w} \tag{66}$$

$$\tilde{v}_{2f}^{\ell} + \tilde{\ell}_{3f}^{\ell} - \tilde{\ell}_{2b}^{w} = \ell_{2b}^{w} \tag{67}$$

- Constraints corresponding to vertices $[u, i, f]$ and $[d, i, b]$

$$\tilde{r}_{i-1,\alpha}^{w} + \tilde{\ell}_{i+1,\alpha}^{w} - \left(\tilde{t}_{i\alpha} + \tilde{e}_{i1\alpha} + \tilde{e}_{i3\alpha} \right) = t_{i\alpha} + e_{i1\alpha} + e_{i3\alpha} \quad \forall (i,\alpha) \in (I \setminus \{2, m\}) \times F \tag{68}$$

$$\tilde{r}_{m-1,\alpha}^{w} - \left(\tilde{t}_{m\alpha} + \tilde{e}_{m1\alpha} + \tilde{e}_{m3\alpha} \right) = t_{m\alpha} + e_{m1\alpha} + e_{m3\alpha} \quad \forall \alpha \in F \tag{69}$$

$$\tilde{y}_u^0 + \tilde{\ell}_{2f}^{w} - \left(\tilde{t}_{1f} + \tilde{e}_{11f} + \tilde{e}_{13f} \right) = t_{1f} + e_{11f} + e_{13f} \tag{70}$$

$$\tilde{\ell}_{2b}^{w} - \left(\tilde{t}_{1b} + \tilde{e}_{11b} + \tilde{e}_{13b} \right) = t_{1b} + e_{11b} + e_{13b} \tag{71}$$

- Constraints corresponding to vertices [u, i, 4] and [d, i, 4]

$$\tilde{w}_{i3\alpha} - \left(\tilde{v}^r_{i\alpha} + \tilde{v}^\ell_{i\alpha}\right) = v^r_{i\alpha} + v^\ell_{i\alpha} \quad \forall\,(i,\alpha) \in (I \setminus \{2,m\}) \times F \tag{72}$$

$$\tilde{w}_{m3\alpha} - \tilde{v}^\ell_{m\alpha} = v^\ell_{m\alpha} \quad \forall\,\alpha \in F \tag{73}$$

$$\tilde{w}_{13f} - \left(\tilde{v}^r_{1f} + \tilde{v}^\ell_{1f}\right) = v^r_{1f} + v^\ell_{1f} \tag{74}$$

$$\tilde{w}_{13b} - \tilde{v}^r_{1b} = v^r_{1b} \tag{75}$$

- Constraints corresponding to vertices [u, i, s] and [d, i, s]

$$\tilde{t}^{if} + \tilde{e}_{i1b} - \tilde{w}_{i1f} = w_{i1f} \quad \forall\,i \in I \tag{76}$$

$$\tilde{t}^{ib} + \tilde{e}_{i1f} - \tilde{w}_{i1b} = w_{i1b} \quad \forall\,i \in I \tag{77}$$

$$\tilde{w}_{i1\alpha} - \tilde{w}_{i2\alpha} = w_{i2\alpha} \quad \forall\,(i,\alpha) \in I \times F \tag{78}$$

$$\tilde{w}_{i2f} + \tilde{e}_{i3b} - \tilde{w}_{i3f} = w_{i3f} \quad \forall\,i \in I \tag{79}$$

$$\tilde{w}_{i2b} + \tilde{e}_{i3f} - \tilde{w}_{i3b} = w_{i3b} \quad \forall\,i \in I \tag{80}$$

- Constraints to link variables

$$\tilde{r}^r_{i\alpha} \le M \cdot r^r_{i\alpha} \quad \forall\,(i,\alpha) \in (I \setminus \{m-1,m\}) \times F \tag{81}$$

$$\tilde{r}^w_{i\alpha} \le M \cdot r^w_{i\alpha} \quad \forall\,(i,\alpha) \in (I \setminus \{m\}) \times F \tag{82}$$

$$\tilde{\ell}^\ell_{i\alpha} \le M \cdot \ell^\ell_{i\alpha} \quad \forall\,(i,\alpha) \in ((I \setminus \{1,2\}) \times F) \cup \{(2,f)\} \tag{83}$$

$$\tilde{\ell}^w_{i\alpha} \le M \cdot \ell^w_{i\alpha} \quad \forall\,(i,\alpha) \in (I \setminus \{1\}) \times F \tag{84}$$

$$\tilde{e}_{is\alpha} \le M \cdot e_{is\alpha} \quad \forall\,(i,s,\alpha) \in I \times \{1,3\} \times F \tag{85}$$

$$\tilde{t}_{i\alpha} \le M \cdot t_{i\alpha} \quad \forall\,(i,\alpha) \in I \times F \tag{86}$$

$$\tilde{w}_{is\alpha} \le M \cdot w_{is\alpha} \quad \forall\,(i,s,\alpha) \in I \times \{1,2,3\} \times F \tag{87}$$

$$\tilde{v}^r_{i\alpha} \le M \cdot v^r_{i\alpha} \quad \forall\,(i,\alpha) \in (I \setminus \{m\}) \times F \tag{88}$$

$$\tilde{v}^\ell_{i\alpha} \le M \cdot v^\ell_{i\alpha} \quad \forall\,(i,\alpha) \in ((I \setminus \{1\}) \times F) \cup \{(1,f)\} \tag{89}$$

$$\tilde{y}^0_\alpha \le M \cdot y^0_\alpha \quad \forall\,\alpha \in \{l,r,u\} \tag{90}$$

Depot inclusion constraint:

$$y^0_l + y^0_r + y^0_u \ge 1 \tag{91}$$

Item inclusion constraints:

$$c^e_{i3f} \cdot \left(w_{i1f} + w_{i3b}\right) \ge b_i \cdot c^e_{i3f} \quad \forall\,i \in I \tag{92}$$

$$c^e_{i3b} \cdot \left(w_{i1b} + w_{i3f}\right) \ge b_i \cdot c^e_{i3b} \quad \forall\,i \in I \tag{93}$$

Constraints for the domains of the variables:

$$r^r_{i\alpha} \in \{0,1\} \quad \forall\,(i,\alpha) \in (I \setminus \{m-1,m\}) \times F \tag{94}$$

$$r^w_{i\alpha} \in \{0,1\} \quad \forall\,(i,\alpha) \in (I \setminus \{m\}) \times F \tag{95}$$

$$\ell^\ell_{i\alpha} \in \{0,1\} \quad \forall\,(i,\alpha) \in ((I \setminus \{1,2\}) \times F) \cup \{(2,f)\} \tag{96}$$

$$\ell^w_{i\alpha} \in \{0,1\} \quad \forall\,(i,\alpha) \in (I \setminus \{1\}) \times F \tag{97}$$

$$e_{is\alpha} \in \{0,1\} \quad \forall\,(i,s,\alpha) \in I \times \{1,3\} \times F \tag{98}$$

$$t_{i\alpha} \in \{0,1\} \quad \forall\,(i,\alpha) \in I \times F \tag{99}$$

$$w_{is\alpha} \in \{0,1\} \quad \forall\,(i,s,\alpha) \in I \times \{1,2,3\} \times F \tag{100}$$

$$v^r_{i\alpha} \in \{0,1\} \quad \forall\,(i,\alpha) \in (I \setminus \{m\}) \times F \tag{101}$$

$$v^\ell_{i\alpha} \in \{0,1\} \quad \forall\,(i,\alpha) \in ((I \setminus \{1\}) \times F) \cup \{(1,f)\} \tag{102}$$

$$y^0_\alpha \in \{0,1\} \quad \forall\,\alpha \in \{l,r,u\} \tag{103}$$

$$y^\ell_0 \in \{0,1\} \tag{104}$$

$$\tilde{r}^r_{i\alpha} \ge 0 \quad \forall\,(i,\alpha) \in (I \setminus \{m-1,m\}) \times F \tag{105}$$

$$\tilde{r}^w_{i\alpha} \ge 0 \quad \forall\,(i,\alpha) \in (I \setminus \{m\}) \times F \tag{106}$$

$$\tilde{\ell}^\ell_{i\alpha} \ge 0 \quad \forall\,(i,\alpha) \in ((I \setminus \{1,2\}) \times F) \cup \{(2,f)\} \tag{107}$$

$$\tilde{\ell}^w_{i\alpha} \ge 0 \quad \forall\,(i,\alpha) \in (I \setminus \{1\}) \times F \tag{108}$$

$$\tilde{e}_{is\alpha} \ge 0 \quad \forall\,(i,s,\alpha) \in I \times \{1,3\} \times F \tag{109}$$

$$\tilde{t}_{i\alpha} \ge 0 \quad \forall\,(i,\alpha) \in I \times F \tag{110}$$

$$\tilde{w}_{is\alpha} \ge 0 \quad \forall\,(i,s,\alpha) \in I \times \{1,2,3\} \times F \tag{111}$$

$$\tilde{v}^r_{i\alpha} \ge 0 \quad \forall\,(i,\alpha) \in (I \setminus \{m\}) \times F \tag{112}$$

$$\tilde{v}^\ell_{i\alpha} \ge 0 \quad \forall\,(i,\alpha) \in ((I \setminus \{1\}) \times F) \cup \{(1,f)\} \tag{113}$$

$$\tilde{y}^0_\alpha \ge 0 \quad \forall\,\alpha \in \{l,r,u\} \tag{114}$$

References

Bozer, Y. A., & Kile, J. W. (2008). Order batching in walk-and-pick order picking systems. *International Journal of Production Research, 46*, 1887–1909.

Burkard, R., Deneko, V. G., van der Veen, J. A. A., & Woeginger, G. J. (1998). Well-solvable special cases of the traveling salesman problem: A survey. *SIAM Review, 40*, 496–546.

Claus, A. (1984). A new formulation for the travelling salesman problem. *SIAM Journal on Algebraic and Discrete Methods, 5*, 21–25.

Dantzig, G. B., Fulkerson, D. R., & Johnson, S. M. (1954). Solution of a large-scale traveling salesman problem. *Journal of the Operations Research Society of America, 2*, 363–410.

Filip, E., & Otakar, M. (2011). The travelling salesman problem and its application in logistic practice. *WSEAS Transactions on Business and Economics, 8*, 163–173.

Gavish, B., & Graves, S. C. (1978). *The traveling salesman problem and related problems. Working Paper GR-078-78*. Operations Research Center, Massachusetts Institute of Technology.

Glover, F., & Punnen, A. P. (1997). The travelling salesman problem: New solvable cases and linkages with the development of approximation algorithms. *Journal of the Operational Research Society, 48*, 502–510.

Gouveia, L., & Pires, J. M. (2001). The asymmetric travelling salesman problem: On generalizations of disaggregated Miller–Tucker–Zemlin constraints. *Discrete Applied Mathematics, 112*, 129–145.

Gu, J., Goetschalckx, M., & McGinnis, L. F. (2007). Research on warehouse operation: A comprehensive review. *European Journal of Operational Research, 177*, 1–21.

Henn, S., Koch, S., Dörner, K. F., Strauss, C., & Wäscher, G. (2010). Metaheuristics for the order batching problem in manual order picking systems. *BuR – Business Research, 3*, 82–105.

Henn, S., & Wäscher, G. (2012). Tabu search heuristics for the order batching problem in manual order picking systems. *European Journal of Operational Research, 222*, 484–494.

Hong, S., Johnson, A. L., & Peters, B. A. (2012). Batch picking in narrow-aisle order picking systems with consideration for picker blocking. *European Journal of Operational Research, 221*, 557–570.

Jarvis, J. M., & McDowell, E. D. (1991). Optimal product layout in an order picking warehouse. *IIE Transactions, 23*, 93–102.

de Koster, R., Le-Duc, T., & Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review. *European Journal of Operational Research, 182*, 481–501.

de Koster, R., & van der Poort, E. (1998). Routing orderpickers in a warehouse: A comparison between optimal and heuristic solutions. *IIE Transactions, 30*, 469–480.

Lenstra, J. K., & Rinnooy Kan, A. H. G. (1975). Some simple applications of the travelling salesman problem. *Operational Research Quarterly, 26*, 717–733.
Letchford, A. N., Nasiri, S. D., & Theis, D. O. (2013). Compact formulations of the Steiner traveling salesman problem and related problems. *European Journal of Operational Research, 228*, 83–92.
Matai, R., Singh, S. P., & Mittal, M. L. (2010). Traveling salesman problem: An overview of applications, formulations and solution approaches. In D. Davendra (Ed.), *Traveling salesman problem: Theory and applications* (pp. 1–24). InTech.
Miller, C. E., Tucker, A. W., & Zemlin, R. A. (1960). Integer programming formulations and traveling salesman problems. *Journal of the Association for Computing Machinery, 7*, 326–329.
Öncan, T., Altinel, K., & Laporte, G. (2009). A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research, 36*, 637–654.
Padberg, M., & Sung, T. (1991). An analytical comparison of different formulations of the traveling salesman problem. *Mathematical Programming, 52*, 315–357.
Petersen, C. G., & Schmenner, R. W. (1999). An evaluation of routing and volume-based storage policies in an order picking operation. *Decision Science, 30*, 481–501.
Ratliff, H. D., & Rosenthal, A. R. (1983). Order-picking in a rectangular warehouse: A solvable case of the traveling salesman problem. *Operations Research, 31*, 507–521.
Rego, C., Gamboa, D., Glover, F., & Osterman, C. (2011). Traveling salesman problem heuristics: Leading methods, implementations and latest advances. *European Journal of Operational Research, 211*, 427–441.
Roodbergen, K. J. (2001). *Layout and routing methods for warehouses*. Rotterdam: Trial.
Roodbergen, K. J., & de Koster, R. (2001a). Routing order pickers in a warehouse with a middle aisle. *European Journal of Operational Research, 133*, 32–43.
Roodbergen, K. J., & de Koster, R. (2001b). Routing methods for warehouses with multiple cross aisles. *International Journal of Production Research, 39*, 1865–1883.
Tompkins, J. A., White, J. A., Bozer, Y. A., & Tanchoco, J. M. A. (2010). *Facilities planning* (4th ed.). New Jersey: John Wiley & Sons.
Wäscher, G. (2004). Order picking: A survey of planning problems and methods. In H. Dyckhoff, R. Lackes, & J. Reese (Eds.), *Supply chain management and reverse logistics* (pp. 324–370). Berlin: Springer.

Part IV:

Picker Routing in Standard-Aisle, Multi-Block Warehouses

An Exact Solution Approach to the Single-Picker Routing Problem in Warehouses with Multiple Blocks

A. Scholz

Faculty of Economics and Management, Otto-von-Guericke University Magdeburg, Germany

Abstract

The Single-Picker Routing Problem (SPRP) deals with the determination of the sequence according to which requested items are to be retrieved from their storage locations in the warehouse. The arrangement of the storage locations typically constitutes a so-called block layout. Based on this structure, exact algorithms have been developed which solve practical-sized SPRP instances within fractions of a second. However, these algorithms are restricted to SPRPs in a single- and a two-block layout. No efficient algorithm exists so far which is able to deal with more than two blocks. In this paper, a model formulation to the SPRP in a multi-block layout is proposed, and different procedures are presented in order to significantly reduce the size of the model. By means of extensive numerical experiments, the impact of these procedures on the size of the model as well as on the computing time required for the generation of an optimal solution is investigated. It is shown that the model can be used to solve very large SPRP instances within a small amount of computing time. Furthermore, it is observed that the computing time does not increase with an increasing number of blocks.

Keywords: Picker Routing, Traveling Salesman, Order Picking

Corresponding author:

André Scholz

Postbox 4120, 39016 Magdeburg, Germany

Phone: +49 391 67 51841

Fax: +49 391 67 48223

Email: andre.scholz@ovgu.de

1 Introduction

Every day, a warehouse receives a high number of items in large lot sizes which have to be stored and redistributed in small volumes based on thousands of daily customer orders (Wäscher, 2004). The retrieval of requested items from their storage locations (order picking) accounts for up to 55% of the costs in a warehouse (Tompkins et al., 2010), which can be attributed to the fact that, in many warehouses, human operators (order pickers) are assigned to execute the picking process (de Koster et al., 2007). This process is mainly composed of traveling through the warehouse, searching for the respective items and picking them from their storage locations, while traveling consumes approximately 50% of the total working time of a picker (Tompkins et al., 2010). In order to reduce the travel time, different procedures can be applied which are improving the allocation of the articles in the warehouse (storage assignment), grouping customer orders into picking orders (order batching) and determining a sequence according to which the picker retrieves the items (picker routing).

The last-mentioned procedure is a part of the so-called Single-Picker Routing Problem (SPRP), which deals with finding a tour of minimum length including all storage locations of requested items (Ratliff & Rosenthal, 1983). The SPRP represents a special case of the Traveling Salesman Problem (TSP) and, thus, approaches to the TSP can be applied to solve the SPRP. However, the storage locations in the warehouse are typically arranged in such a way that they constitute a block layout (Roodbergen, 2001). This fact is totally neglected when modeling the SPRP as a general TSP, which mainly results in two problems. First, problem-specific approaches to the SPRP may outperform TSP approaches by far in terms of computing time. Second, TSP approaches may not be able to deal with additional constraints arising in practical applications. For example, order pickers seem to prefer simple routes which are easy to memorize (Petersen & Schmenner, 1999). Furthermore, some aisles may be very narrow making it impossible for the order picker to change his moving direction within the aisle. The possibility of depositing retrieved items at the end of each picking aisle represents another modification often arising in practice (de Koster & van der Poort, 1998). These constraints can easily be taken into account by problem-specific approaches, whereas their integration into TSP algorithms may not be straightforward.

Efficient problem-specific solution approaches have been proposed by Ratliff & Rosenthal (1983) and Roodbergen & de Koster (2001) for the case of a single- and a two-block layout, respectively. However, no efficient algorithm exists which can deal with more than two blocks (Roodbergen, 2001). Therefore, in this paper, a problem-specific model formulation to the SPRP is developed, which can be used to optimally solve large-sized SPRP instances with an arbitrary number of blocks. The model formulation represents an extension of the model of Scholz et al. (2016), which is meant to solve SPRPs in a single-block layout. Since the number of variables and constraints rapidly increases with an increasing number of blocks, the underlying graph is modified in order to keep the size of the model at a reasonable level. First, a so-called pyramid structure is introduced, which cuts off the components of the graph representing parts of the warehouse not included in an optimal tour. In a second step, the number of vertices and arcs required for the representation of the locations of the items is reduced.

The modifications result in a model formulation whose size is not only dependent on the size of the problem, i.e. on the number of requested items and on the size of the warehouse, but it also depends on the respective storage locations of the requested items. Due to this fact, it is not possible to determine the size of the model for a given class of problem instances without conducting numerical experiments. Thus, the impact of the modifications cannot be evaluated in advance and no conclusions can be drawn whether application of the modifications to a certain instance is a worthwhile endeavor. Therefore, formulas are developed in order to estimate the number of variables and constraints included in the model. By means of numerical experiments, it is shown that quite good estimations are obtained by using the formulas, and problem classes are pointed out where the application of the modifications is inevitable for solving instances within a reasonable amount of computing time. Furthermore, the results of the experiments clearly demonstrate that the model formulation is able to deal with any practical-sized problem instance, while computing times do not increase with an increasing number of blocks.

The remainder of this paper is organized as follows: The SPRP is introduced and the related literature is briefly reviewed in the next section. In Section 3, a graph representing the SPRP is first constructed based on the suggestions of Scholz et al. (2016). Different conditions are then investigated under which the size of the graph can be reduced significantly, and formulas are developed in order to estimate the number of vertices and arcs included in the modified graph. The components of the resulting model formulation are described in Section 4, while Section 5 comprises the design of the numerical experiments and the results obtained from them. The paper concludes with an outlook on further research (Section 6).

2 Single-Picker Routing Problem

2.1 Problem description

The SPRP consists of finding a tour through the warehouse which starts and ends at the depot while all requested items are retrieved. In the warehouse, the items are stored on pallets or racks which typically constitute a block layout (Roodbergen, 2001). According to this layout, two different types of aisles have to be distinguished, namely picking aisles and cross aisles. Picking aisles are arranged parallel to each other and include the storage locations of the items. Thus, for retrieving an item, the corresponding picking aisle has to be entered. In contrast, cross aisles do not contain any storage locations, but they are required for changing over from one picking aisle to another. Furthermore, cross aisles divide the picking area of the warehouse into blocks and picking aisles into subaisles. A block is the part of the picking area located between two adjacent cross aisles, and a subaisle is defined as the part of a picking aisle corresponding to the same block. A warehouse with $q+1$ cross aisles and m picking aisles consists of q blocks and $q \cdot m$ subaisles. The corresponding layout is called a q-block layout.

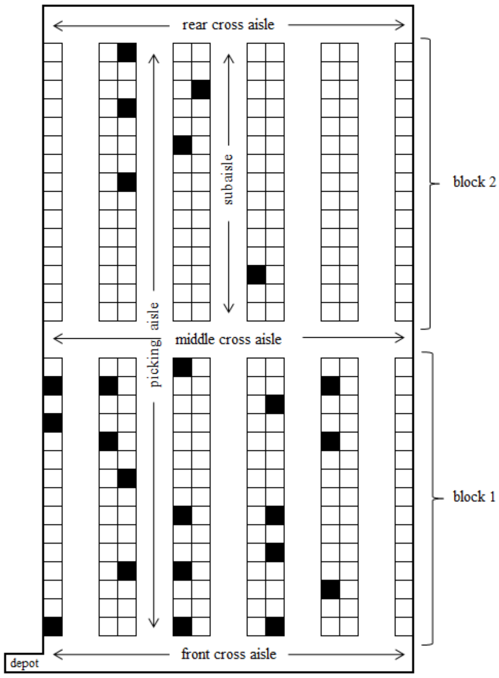


Fig. 1: Two-block layout

In Fig. 1, a picking area with two blocks and 5 picking aisles is depicted. The rectangles symbolize the storage locations and the black rectangles represent the locations of the requested items (pick locations). The depot is situated in the bottom left-hand corner of the picking area. Three cross aisles exist, namely the front, a middle and the rear cross aisle, which can be used to switch between picking aisles. The front (rear) cross aisle represents the cross aisle nearest to (farthest from) the depot. In order to change over from one block to another, the middle cross aisle has to be crossed.

The retrieval of requested items from their storage locations is carried out by an order picker who walks or rides through the warehouse using a picking device. Among all operations required for retrieving items and returning them to the depot, traveling consumes a major part of the working time of an order picker (Tompkins et al., 2010). Furthermore, other components (such as setup times at the depot or searching and picking times at the pick locations) can be considered to be constants (Caron et al., 2000) because they are independent of the sequence according to which the items are to be picked. Therefore, minimizing the total travel time is a common objective for the SPRP. Assuming a constant travel velocity, the travel time is a linearly increasing function of the travel distance (Jarvis & McDowell, 1991), implying that minimizing the travel time is equivalent to finding a tour of minimum length.

The SPRP can then be defined as follows (Ratliff & Rosenthal, 1983; Scholz et al., 2016): Given a set of items to be picked from known storage locations, in which sequence should the locations be visited such that the total length of the corresponding tour is minimized?

2.2 Literature review

Since the SPRP represents a special case of the TSP, general TSP formulations may be suitable for modeling the SPRP. For the first time, Dantzig et al. (1954) proposed a model formulation to the TSP. However, in this model, the number of constraints required for the exclusion of subtours exponentially increases with the number of pick locations as it is the case for several other mathematical programming formulations to the TSP (Gouveia & Pires, 2001). Due to memory restrictions, these formulations cannot be applied to large SPRPs. In contrast, compact formulations, characterized by a polynomial increase of the number of variables and constraints in the number of pick locations, allow for explicitly representing larger SPRPs. A variety of compact formulations to the TSP exists (Öncan et al., 2009). However, due to the quality of the lower bounds obtained by solving the corresponding LP relaxations, which may be much weaker than the lower bounds in the Dantzig formulation (Padberg & Sung, 1991), compact TSP formulations may not be able to deal with larger SPRP instances either (Letchford et al., 2013; Scholz et al., 2016).

A more appropriate way to model the SPRP has been presented by Burkard et al. (1998) who formulated the SPRP as a Steiner TSP. In a Steiner TSP, the set of vertices V can be divided into a subset P and a subset $V \setminus P$ (Steiner points). A Steiner tour has to include all vertices of the subset P. Steiner points may but do not have to be visited. Furthermore, vertices and edges are allowed to be visited and used more than once. As for the SPRP, the set P includes the pick locations and the location of the depot. The set of Steiner points comprises the vertices representing the intersections between a cross aisle and a picking aisle. Edges are then introduced between adjacent pick locations situated in the same subaisle, between the pick locations nearest to a cross aisle and the adjacent Steiner point, and between Steiner points corresponding to adjacent intersections. This kind of representation requires far fewer edges than the corresponding TSP graph would include. Since the number of variables is only dependent on the number of edges in many TSP formulations, the consideration of a sparse graph can be expected to result in model formulations of smaller size. Letchford et al. (2013) developed compact formulations to the Steiner TSP, and they demonstrated that the application of these formulations outperforms general TSP formulations in terms of computing time. This observation also holds when solving the SPRP. However, since the size of Steiner TSP formulations is dependent on both the number of pick locations and the number of intersections, these formulations are not suitable for solving SPRPs in large warehouses with many items to be retrieved (Scholz et al., 2016). Furthermore, Steiner TSP as well as general TSP solution approaches do not allow for the integration of problem-specific modifications, such as the construction of simple tours, in which the number of changes in direction is limited or where each subaisle is allowed to be visited once only (Scholz et al., 2016). For the consideration of such aspects, the application of problem-specific approaches to the SPRP is inevitable.

For warehouses with one or two blocks, efficient problem-specific solution approaches exist. Ratliff & Rosenthal (1983) proposed an exact algorithm for the SPRP in a single-block layout which is based on dynamic programming. The computational effort of the algorithm increases linearly with the number of pick locations and picking aisles and, therefore, it can be used to solve any practical-sized instance within fractions of a second. Roodbergen & de Koster (2001) extended this algorithm to the two-block case. However, the authors stated that it would be very difficult to further extend this approach to layouts with more than two blocks. The model formulation of Scholz et al. (2016) is the only problem-specific exact solution approach which can be extended straightforwardly to the case of multiple blocks. Based on the representation of the SPRP as a Steiner TSP and some characteristics of optimal SPRP solutions, a graph to the SPRP in a single-block layout was first constructed whose size is independent of the number of pick locations. A model formulation was then obtained by application of a TSP formulation to the underlying graph. By means of numerical experiments, Scholz et al. (2016) demonstrated that general and Steiner TSP formulations are outperformed by far in terms of computing time. The authors also briefly described how the model could be extended to SPRPs in multiple blocks. However, they noted that the size of the graph, as well as the size of the model, would significantly increase, which will lead to serious computing time issues if several blocks have to be dealt with.

According to the suggestions of Scholz et al. (2016), the model formulation is extended to the case of multiple blocks in the next section. Modifications to the graph are then proposed in order to keep the size of the graph and the resulting model formulation at a reasonable level even for large SPRP instances.

3 Representation of the SPRP in a multi-block layout

3.1 Graph construction according to Scholz et al. (2016)

As mentioned before, the underlying graph of the model formulation of Scholz et al. (2016) is based on the Steiner TSP representation of the SPRP. A directed graph is considered, i.e. each edge is replaced by two reverse arcs representing possible movements of an order picker. Arcs between Steiner points relate to movements within cross aisles, whereas the other arcs (except for arcs incident to the vertex representing the depot) stand for movements in subaisles. The latter can be further restricted due to the structure of optimal solutions to the SPRP. In an optimal tour, the following movements can be performed for retrieving all requested items in a subaisle (Ratliff & Rosenthal, 1983):

(a) the subaisle is entered from an adjacent cross aisle and left via the other adjacent cross aisle;

(b) the subaisle is entered and left via the same adjacent cross aisle, while the picker returns at the pick location farthest from this cross aisle;

(c) the subaisle is visited twice in such a way that the largest gap, i.e. the largest distance between two adjacent pick locations or a pick location and the adjacent cross aisle, is skipped.

The pick locations of a subaisle can then be represented by using four vertices only. One vertex is required for each pick location which is adjacent to a cross aisle while the remaining two vertices correspond to the pick locations defining the largest gap. This results in a graph whose size is independent of the number of pick locations. In order to ensure that each vertex is visited at most once, Scholz et al. (2016) modified the graph by copying the vertices based on the maximum number of visits in an optimal tour. One copy is introduced for each vertex representing a pick location, two copies are generated for each front and rear cross aisle vertex and three copies are inserted for each vertex which represents an intersection with the middle cross aisle. An example graph is depicted in Fig. 2.

Vertex "0" represents the location of the depot. The denotation of the other vertices is as follows: The first entry indicates the direction in which the tour is proceeded after visiting the vertex, where "r" and "l" symbolize that the next step will be a movement to the right and to the left, respectively. Movements towards the rear and the front cross aisle are indicated by "u" ("up") and "d" ("down"). The second component stands for the number of the block if the vertex represents a pick location, or the number of the cross aisle, otherwise. Cross aisles are enumerated from 1 to $p+1$, where p is the number of blocks and cross aisle 1 is the cross aisle nearest to the depot. The third component characterizes the number of the corresponding picking aisle. Picking aisles are enumerated in ascending order from left to right, i.e.

picking aisle 1 is the leftmost picking aisle while m denotes the rightmost aisle. Furthermore, vertices representing a pick location have an additional fourth component indicating the position of the vertex in the corresponding subaisle. The arcs have been introduced according to the possible movements in a subaisle. If the arc $([u,q,1],[u,q,i,1])$ or $([d,q,1],[d,q,i,1])$ is visited first, then the subaisle has to be traversed. The arcs $([u,q,1],[d,q,i,1])$ and $([d,q,1],[u,q,i,1])$ indicate that the subaisle is entered and left via the same cross aisle, while the arcs $([u,q,1],[d,q,i,3])$ or $([d,q,1],[u,q,i,3])$ symbolize that the largest gap is skipped.

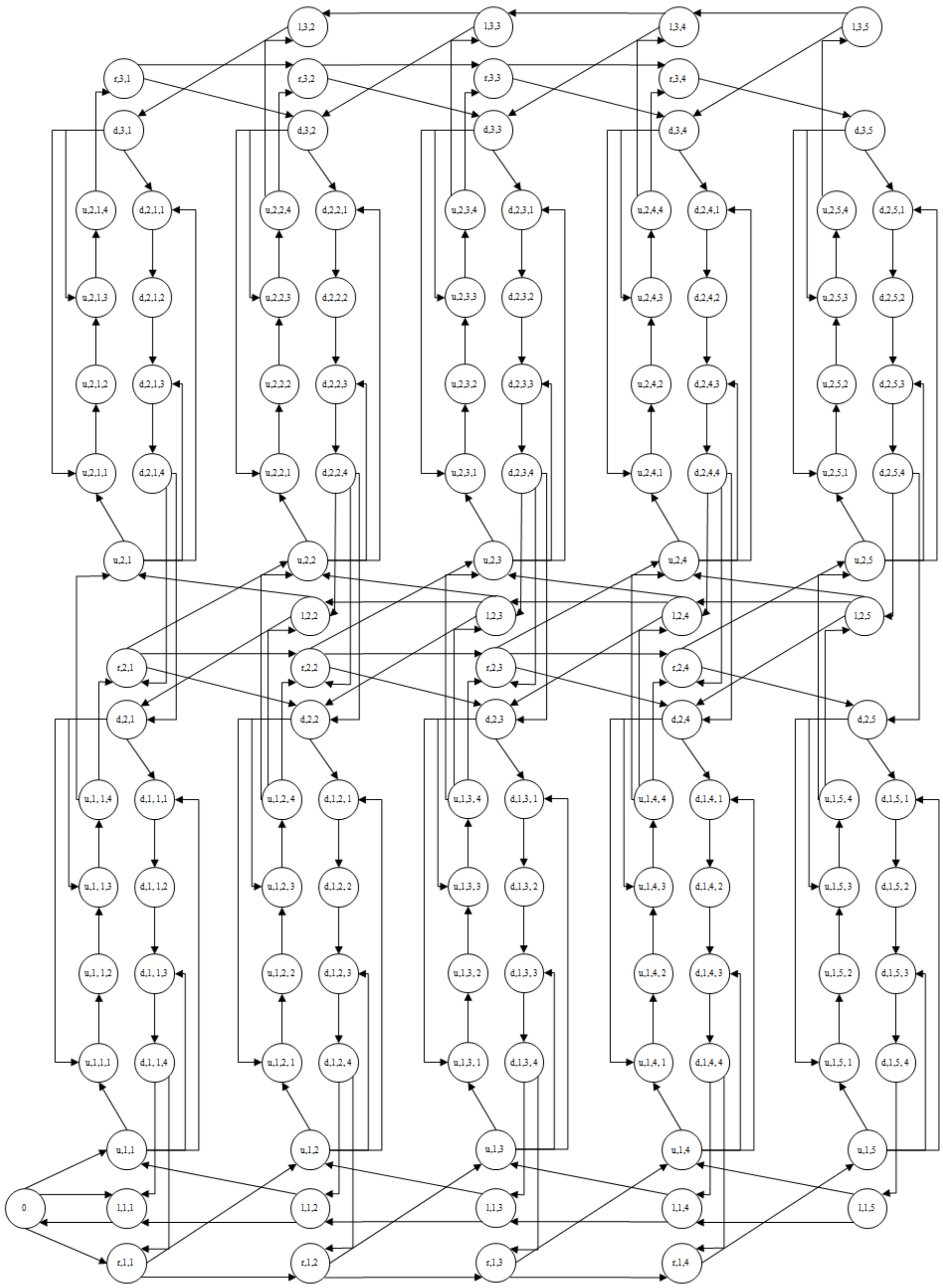


Fig. 2: Graph for a SPRP with two blocks and five picking aisles

3.2 Reduction of the number of subaisles to be represented

The graph presented in the previous subsection allows for the representation of the SPRP using a constant number of vertices per subaisle, which is advantageous if the number of pick locations is quite large compared to the number of subaisles. However, in case of multiple blocks, this kind of representation becomes the major drawback of the graph. If multiple blocks have to be dealt with, the number of subaisles significantly increases, resulting in far fewer pick locations per subaisle. In particular, if using class-based storage assignment procedures (Petersen & Schmenner, 1999), it will be most likely that lots of subaisles will not contain any pick location at all. Nevertheless, each subaisle will be included in the graph and represented by the same number of vertices and arcs. In order to reduce the size of the graph and overcome this drawback, in the following, it is investigated under which conditions a subaisle can be removed from the graph without affecting the minimal tour length.

Let $B = \{1,\dots,p\}$ be the set of blocks and $\overline{\mathcal{M}}_q$ the rightmost subaisle of block $q \in B$ containing at least one requested item. Furthermore, $\overline{m}_q$ denotes the rightmost subaisle of block $q \in B$ to be included in the graph in order to not affect the minimal tour length. Obviously, it must hold $\overline{m}_q \geq \overline{\mathcal{M}}_q$ for each block $q \in B$ since each subaisle containing a requested item has to be visited. Another reason for visiting a subaisle is to change over to another cross aisle in order to reach an adjacent block. Due to this fact, simply removing all subaisles containing no pick location could result in longer tours. Thus, it is necessary to also take the characteristics of other blocks into account. In fact, a subaisle of a block $q \in B$ is not removed if the adjacent upper block, i.e. a block farther from the depot, includes a subaisle, which has to be considered and is located further on the right.

The index $\overline{m}_q$ of the rightmost subaisle of block $q \in B$ to be included in the graph can then be determined by solving the following mathematical program. (Note that $\overline{m}_{p+1}$ is defined to be 0.)

$$\min \ \overline{m}_q \tag{1}$$

$$\overline{m}_q \geq \overline{\mathcal{M}}_q \tag{2}$$

$$\overline{m}_q \geq \overline{m}_{q+1} \tag{3}$$

The objective function minimizes the index of the rightmost subaisle to be considered in block q. Constraint (2) guarantees that no subaisle of block q containing at least one requested item is removed from the graph. Constraint (3) ensures that the index of the rightmost subaisle to be considered is not larger for block $q+1$ than for block q. Due to this constraint, the resulting structure of the graph is referred to as a pyramid structure.

Since $\overline{m}_{p+1} = 0$, the optimal solution for block p is $\overline{m}_p = \overline{\mathscr{M}}_p$. Then $\overline{m}_q$ can be determined successively for the remaining blocks $q \in B \setminus \{p\}$:

$$\overline{m}_q = \max\left\{\overline{\mathscr{M}}_q;\ \overline{m}_{q+1}\right\}. \tag{4}$$

An analogue procedure can be applied to subaisles located in the first (leftmost) subaisles of a block. Let $\underline{\mathscr{M}}_q$ be the leftmost subaisle of block $q \in B$ containing a pick location and $\underline{m}_q$ the leftmost subaisle of block $q \in B$ which has to be considered for the construction of an optimal tour. With the same line of argumentation as above, it results $\underline{m}_p = \underline{\mathscr{M}}_p$ and for each block $q \in B \setminus \{1, p\}$:

$$\underline{m}_q = \min\left\{\underline{\mathscr{M}}_q;\ \underline{m}_{q+1}\right\}. \tag{5}$$

Since the depot is assumed to be situated in the bottom left-hand corner of the picking area, $\underline{m}_1$ has to be set to 1 even if the leftmost picking aisle does not include any pick locations.

For each block $q \in B$, a subaisle is then removed if it is located further to the left than subaisle $\underline{m}_q$ or in the right of subaisle $\overline{m}_q$. When removing a subaisle i of a block q, all of the eight vertices representing the pick locations are deleted. Since this aisle will not be entered in an optimal tour, vertices $[u, i, q]$ and $[d, i, q+1]$ will be removed as well. Furthermore, vertices $[r, i-1, q+1]$ and $[l, i, q+1]$, required for switching over to the subaisle, are also deleted. In conjunction with these vertices, incident arcs will be removed. In total, dependent on the location of the subaisle, a removal of a single subaisle will reduce the size of the graph by up to 12 vertices and 30 arcs.

3.3 Improvements related to the representation of a subaisle

In the graph of Scholz et al. (2016), independent of the number of pick locations included, each subaisle is represented by 8 vertices, while 18 arcs are required to specify the possible movements within the subaisle (see Fig. 2). If a subaisle contains only a few or even no pick locations, Scholz et al. (2016) introduced dummy vertices and arcs. This approach may cause problems in case of multiple blocks since lots of dummy vertices and arcs will be introduced due to the small ratio between the number of pick locations and the number of subaisles. Therefore, special cases are identified for the locations of the requested items of a subaisle, which allow for removing vertices and arcs. For the sake of simplicity of exposition, a subaisle i of a block $q \in B \setminus \{1, p\}$ with $\underline{m}_{q+1} < i < \overline{m}_{q+1}$ is considered.

The largest reduction can be observed when a subaisle does not contain any pick locations (denoted by special case 1). In this case, arcs are only required in order to ensure that the subaisle can be used to switch over to an adjacent cross aisle. On the left-hand side of Fig. 3, the subaisle including the storage locations is depicted, while the corresponding part of the graph is shown on the right-hand side. For entering the subaisle, either vertex $[u,q,i]$ or $[d,q+1,i]$ has to be visited. The order picker can then proceed the tour by going to the left or to the right or by entering a subaisle of an adjacent block. The representation of the above-mentioned movements requires 6 arcs, resulting in a reduction of 8 vertices (100%) and 12 arcs (67%) per subaisle.

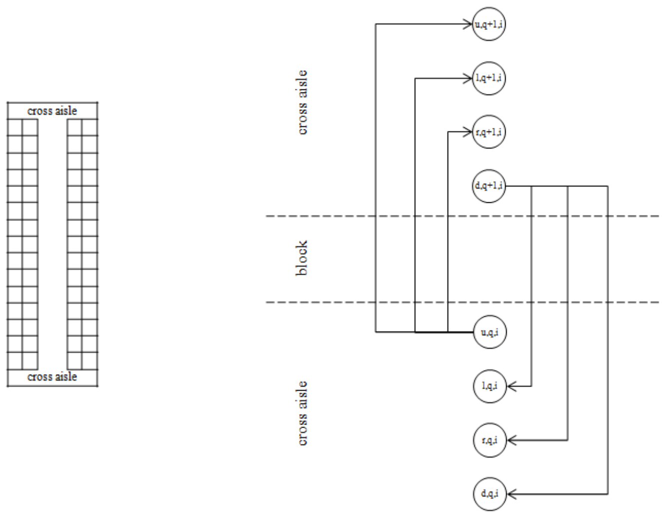


Fig. 3: Representation of a subaisle containing no pick locations

Another considerable size reduction can be obtained if the largest gap lies between cross aisle q and the adjacent pick location (special case 2). Vertex $[u,q,i]$ and vertex pair $([u,q,i,2],[d,q,i,3])$ will then represent the same location. In this case, skipping the largest gap is the same as entering and leaving the subaisle via cross aisle $q+1$. For the representation of such a subaisle, two vertices $[u,q,i,1]$ and $[d,q,i,1]$ are required, which refer to the pick locations nearest to and farthest from the depot (see Fig. 4). Note that these vertices do not form a vertex pair, i.e. the distance to vertex $[u,q,i]$ may be different for both vertices. Here, 2 instead of 8 vertices (75% reduction) and 10 instead of 18 arcs are needed (44% reduction). The same line of argumentation holds if the largest gap lies between cross aisle $q+1$ and the adjacent pick location.

A very simple possibility to reduce the size of the graph arises when only two pick locations are contained in a subaisle, while the largest gap is situated between these locations (special case 3). A vertex pair is then introduced for each pick location and the arcs are chosen in such a way that all movements can be performed. 4 vertices (50% reduction) and 14 arcs (22% reduction) are needed to represent the locations and the movements in the subaisle.

If the largest gap is given by two pick locations and one of them is adjacent to the lower cross aisle, a slight size reduction can be achieved, since the vertex pairs $([u,q,i,1],[d,q,i,4])$ and $([u,q,i,2],[d,q,i,3])$

define the same location (special case 4). Therefore, one vertex pair can be removed from the graph, resulting in a reduction of 2 vertices (25%) and 2 arcs (11%). The same reduction is obtained if one of the pick locations is adjacent to the upper cross aisle (special case 5).

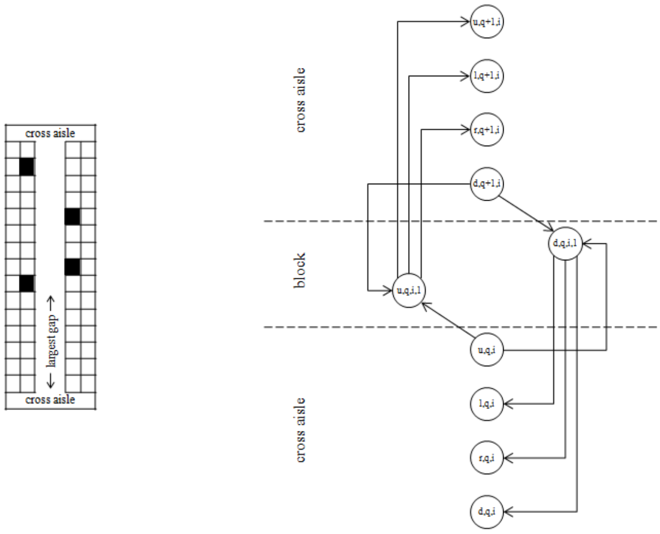


Fig. 4: Representation of a subaisle with the largest gap located between a pick location and the adjacent cross aisle

3.4 Estimation of the size of the reduced graph

The graph depicted in Fig. 2 contains 125 vertices and 217 arcs. After introduction of the pyramid structure and consideration of the special cases of item distribution, the number of vertices and arcs amounts to 54 and 98, respectively, which corresponds to a reduction of the size of the graph by approximately 55%. In general, the amount of reduction obtained is dependent on the size of the respective SPRP instance on the one hand, but it also depends on the actual locations of the requested items on the other hand. Thus, the impact of the reduction procedures on the size of the graph cannot be quantified in advance for problem instances of a certain size. Therefore, formulas are given, which allow for estimating the number of remaining vertices and arcs for an instance with n pick locations, p blocks, m picking aisles and a certain storage assignment policy. The formulas are developed in two steps. First, the vertices representing locations in cross aisles and adjacent arcs are considered. Second, the vertices representing the pick locations and the corresponding adjacent arcs are added to the graph.

In the first step, the representation of the pick locations is not considered any further, i.e. each subaisle is assumed to follow special case 1. Before the removal of subaisles by application of the pyramid structure, the number of vertices v^c and arcs a^c per subaisle is then as follows:

$$v^c = \frac{4pm - 2p + 2m - 2}{pm} = 4 - \frac{2}{m} + \frac{2}{q} - \frac{2}{pm}; \tag{6}$$

$$a^c = \frac{12pm - 12p - 4}{pm} = 12 - \frac{12}{m} - \frac{4}{pm}. \tag{7}$$

Estimations for the number of vertices $E(V^c)$ and for the number of arcs $E(A^c)$ included in the graph after the introduction of the pyramid structure are given by

$$E(V^c) = E(S) \cdot v^c + 2 \text{ and} \tag{8}$$

$$E(A^c) = E(S) \cdot a^c + 6, \tag{9}$$

respectively, where $E(S)$ denotes the expected number of subaisles not removed from the graph. The constant part in equation (8) is related to the vertices "0" and $[l,1,1]$, while the constant term in (9) refers to arcs adjacent to these vertices. For the determination of $E(S)$, the following symbols are introduced:

α_{qi}: probability (given by the storage assignment policy) of an item to be stored in subaisle i of block q;

X_{qi}: number of requested items located in subaisle i of block q;
$X_{qi} \sim B(n, \alpha_{qi})$, i.e. $P(X_{qi} = k) = \binom{n}{k} \cdot \alpha_{qi}^k \cdot (1 - \alpha_{qi})^{n-k}$;

$\overline{Q}$: index of the block to be considered which is farthest from the depot;

δ_q^l: difference between the number of subaisles of block q and block $q+1$, which have been removed on the left-hand side of the blocks; $\delta_q^l = \underline{m}_{q+1} - \underline{m}_q$ for $q \in \{1, \ldots, \overline{Q} - 1\}$ and $\delta_{\overline{Q}}^l = \overline{m}_{\overline{Q}} - \underline{m}_{\overline{Q}} + 1$;

δ_q^r: difference between the number of subaisles of block q and block $q+1$, which have been removed on the right-hand side of the blocks; $\delta_q^r = \overline{m}_{q-1} - \overline{m}_q$ for $q \in \{2, \ldots, \overline{Q}\}$ and $\delta_1^r = m - \overline{m}_1$.

Now, $E(S)$ is given by:

$$E(S) = \sum_{s=1}^{p \cdot m} s \cdot P(S = s) = \sum_{s=1}^{p \cdot m} s \cdot \sum_{q=1}^{p} P\left(S = s \mid \overline{Q} = q\right) \cdot P\left(\overline{Q} = q\right). \tag{10}$$

First, the probability $P\left(\overline{Q} = q\right)$ of block q being the farthest block with a requested item is considered:

$$\begin{aligned} P\left(\overline{Q} = q\right) &= P\left(\sum_{w=q+1}^{p} \sum_{i=1}^{m} X_{wi} = 0,\ \sum_{i=1}^{m} X_{qi} > 0\right) \\ &= P\left(\sum_{w=q+1}^{p} \sum_{i=1}^{m} X_{wi} = 0\right) - P\left(\sum_{w=q+1}^{p} \sum_{i=1}^{m} X_{wi} = 0,\ \sum_{i=1}^{m} X_{qi} = 0\right). \end{aligned} \tag{11}$$

Let $Y = (Y_1, Y_2, Y_3)$ denote the number of requested items included in the first $q-1$ blocks (Y_1), in the q[th] block (Y_2) and in the last $p-q$ blocks (Y_3), respectively. Y follows a multinomial distribution, i.e.:

$$P(Y_1 = k_1, Y_2 = k_2, Y_3 = k_3) = \binom{n}{k_1, k_2, k_3} \cdot \left(\sum_{w=1}^{q-1} \sum_{i=1}^{m} \alpha_{wi}\right)^{k_1} \cdot \left(\sum_{i=1}^{m} \alpha_{qi}\right)^{k_2} \cdot \left(\sum_{w=q+1}^{p} \sum_{i=1}^{m} \alpha_{wi}\right)^{k_3}, \tag{12}$$

where $\binom{n}{k_1,k_2,k_3} = \frac{n!}{k_1!\cdot k_2!\cdot k_3!}$ represents the multinomial coefficient with $k_1+k_2+k_3=n$. This observation leads to the following result for $q>1$:

$$\begin{aligned} P\left(\overline{Q}=q\right) &= P\left(Y_3=0\right) - P\left(Y_2=0, Y_3=0\right) \\ &= \sum_{j=0}^{n} \binom{n}{j, n-j, 0} \cdot \left(\sum_{w=1}^{q-1}\sum_{i=1}^{m} \alpha_{wi}\right)^{j} \cdot \left(\sum_{i=1}^{m} \alpha_{qi}\right)^{n-j} - \left(\sum_{w=1}^{q-1}\sum_{i=1}^{m} \alpha_{wi}\right)^{n} \\ &= \sum_{j=0}^{n-1} \binom{n}{j} \cdot \left(\sum_{w=1}^{q-1}\sum_{i=1}^{m} \alpha_{wi}\right)^{j} \cdot \left(\sum_{i=1}^{m} \alpha_{qi}\right)^{n-j} . \end{aligned} \tag{13}$$

If $q=1$, all requested items are contained in the subaisles of the first block, i.e. it holds:

$$P\left(\overline{Q}=1\right) = \left(\sum_{i=1}^{m} \alpha_{1i}\right)^{n} . \tag{14}$$

In addition to $P\left(\overline{Q}=q\right)$, the probability $P\left(S=s \mid \overline{Q}=q\right)$ of s to be the number of subaisles included in the reduced graph given that q is the farthest block with requested items has to be determined. For this purpose, the set O_s^q is introduced which describes all pyramid structures containing s subaisles in total given that $\overline{Q}=q$. If all requested items are included in the first q blocks, the actual structure of the pyramid can be derived from the values of δ_w^l and δ_w^r ($w \in \{1,\ldots,q\}$). The set O_s^q is given as follows:

$$\begin{aligned} O_s^q = \Bigg\{ \left(\delta_1^l,\ldots,\delta_q^l,\delta_1^r,\ldots,\delta_q^r\right) \in \{0,1,\ldots,m\}^{2q} \,\Bigg|\, &\sum_{w=1}^{q}\left(\delta_w^l+\delta_w^r\right) = m,\ \delta_q^l > 0, \\ &q\cdot m - \sum_{w=1}^{q-1} \delta_w^l \cdot (q-w) - \sum_{w=1}^{q} \delta_w^r \cdot (q-w+1) = s \Bigg\} \end{aligned} \tag{15}$$

By definition, the sum of all parameters δ_w^l and δ_w^r has to be equal to the number of picking aisles m, while δ_q^l has to be positive in order to ensure that block q includes a requested item. The third condition is related to the number of subaisles s still included in the graph after the introduction of the pyramid structure. The maximum number of subaisles equals $q\cdot m$ in a picking area with q blocks and m picking aisles. In the first block, no picking aisles are removed on the left-hand side because of the location of the depot, whereas the reduction amounts to δ_1^r subaisles for the right-hand side. δ_1^l and $\delta_1^r+\delta_2^r$ subaisles are removed from the left-hand and from the right-hand side of the second block, respectively. Following this procedure, the first $\sum_{w=1}^{q-1} \delta_w^l$ and the last $\sum_{w=1}^{q} \delta_w^r$ subaisles are removed from block q.

The pyramid structure defined by $\delta = \left(\delta_1^l,\ldots,\delta_q^l,\delta_1^r,\ldots,\delta_q^r\right)$ implies that the first $\sum_{\tilde{w}=1}^{w-1} \delta_{\tilde{w}}^l$ subaisles of block $w \in \{2,\ldots,q\}$ and the last $\sum_{\tilde{w}=1}^{w} \delta_{\tilde{w}}^r$ subaisles of block $w \in \{1,\ldots,q\}$ do not include any requested items. Moreover, whenever δ_w^l is positive for a block $w \in \{2,\ldots,q-1\}$, subaisle $\sum_{\tilde{w}=1}^{w} \delta_{\tilde{w}}^l + 1$ of block w has to contain at least one requested item. Otherwise, this subaisle would have been removed by application of the pyramid structure. Analogueously, a positive value of δ_w^r for a block $w \in \{1,\ldots,q-1\}$ implies that subaisle $m - \sum_{\tilde{w}=1}^{w} \delta_{\tilde{w}}^r$ includes a requested item. With respect to block q, it can be seen that requested items have to be located in the subaisles $\sum_{\tilde{w}=1}^{q-1} \delta_{\tilde{w}}^l$ and $\sum_{\tilde{w}=1}^{q} \delta_{\tilde{w}}^l$, respectively. These observations result in the following equations.

$$q=1: \; P\left(S=s \mid \overline{Q}=q\right) = \sum_{\delta \in O_s^q} P\Big(X_{1,m-\delta_1^r} > 0,\; X_{1,m-\delta_1^r+1} = 0,\ldots,X_{1,m}=0\Big) \tag{16}$$

$$\begin{aligned} q>1: \; P\left(S=s \mid \overline{Q}=q\right) = \sum_{\delta \in O_s^q} P\Big(&X_{21}=0,\ldots,X_{2,\delta_1^l}=0,\ldots,X_{q1}=0,\ldots,X_{q,\delta_1^l+\ldots+\delta_{q-1}^l}=0,\\ &X_{1,m-\delta_1^r+1}=0,\ldots,X_{1m}=0,\ldots,X_{q,m-\delta_1^r-\ldots-\delta_q^r+1}=0,\ldots,X_{qm}=0,\\ &X_{w,\delta_1^l+\ldots+\delta_{w-1}^l+1}>0 \text{ for all } w \in \{2,\ldots,q-1\} \text{ with } \delta_w^l>0,\\ &X_{w,m-\delta_1^r-\ldots-\delta_w^r}>0 \text{ for all } w \in \{1,\ldots,q-1\} \text{ with } \delta_w^r>0,\\ &X_{q,\delta_1^l+\ldots+\delta_{q-1}^l+1}>0, X_{q,m-\delta_1^r-\ldots-\delta_q^r}>0\Big) \end{aligned} \tag{17}$$

Let now J_δ^0 comprise all subaisles including no requested items and let J_δ^+ contain all subaisles in which at least one requested item has to be located. By application of the inclusion-exclusion rule, we obtain:

$$\begin{aligned} P\left(S=s \mid \overline{Q}=q\right) &= \sum_{\delta \in O_s^q} P\left(X_{wi}=0 \text{ for all } (w,i) \in J_\delta^0,\; X_{wi}>0 \text{ for all } (w,i) \in J_\delta^+\right)\\ &= \sum_{\delta \in O_s^q} \Big(P\left(X_{wi}=0 \text{ for all } (w,i) \in J_\delta^0\right)\\ &\quad -P\left(X_{wi}=0 \text{ for all } (w,i) \in J_\delta^0,\; X_{wi}=0 \text{ for at least one } (w,i) \in J_\delta^+\right)\Big)\\ &= \sum_{\delta \in O_s^q} \sum_{k=0}^{|J_\delta^+|} (-1)^k \sum_{\substack{J \subseteq J_\delta^+\\ |J|=k}} P\left(X_{wi}=0 \text{ for all } (w,i) \in I_\delta^0 \cup J\right)\\ &= \sum_{\delta \in O_s^q} \sum_{k=0}^{|J_\delta^+|} (-1)^k \sum_{\substack{J \subseteq J_\delta^+\\ |J|=k}} P\left(\sum_{(w,i) \in I_\delta^0 \cup J} X_{wi}=0\right)\\ &= \sum_{\delta \in O_s^q} \sum_{k=0}^{|J_\delta^+|} (-1)^k \sum_{\substack{J \subseteq J_\delta^+\\ |J|=k}} \left(1-\sum_{(w,i) \in I_\delta^0 \cup J} \alpha_{wi}\right)^n. \end{aligned} \tag{18}$$

After the introduction of the pyramid structure, special cases of item distribution are considered in order to reduce the number of vertices and arcs for the representation of the movements in the subaisles. The expected number of vertices $E\left(V^{w}\right)$ required for the representation of all pick locations is given by:

$$\begin{aligned} E\left(V^{w}\right) &= \sum_{q=1}^{p}\sum_{i=1}^{m}\sum_{v=1}^{4} 2v \cdot P\left(V_{qi}^{w}=2v\right) \\ &= \sum_{q=1}^{p}\sum_{i=1}^{m}\sum_{v=1}^{4} 2v \cdot \sum_{j=1}^{n} P\left(V_{qi}^{w}=2v \,\middle|\, X_{qi}=j\right) \cdot P\left(X_{qi}=j\right) \\ &= \sum_{q=1}^{p}\sum_{i=1}^{m}\sum_{v=1}^{4} 2v \cdot \sum_{j=1}^{n} P\left(V_{qi}^{w}=2v \,\middle|\, X_{qi}=j\right) \cdot \binom{n}{j} \cdot \alpha_{qi}^{j} \cdot \left(1-\alpha_{qi}\right)^{n-j}. \end{aligned} \tag{19}$$

According to the special cases of item distribution, the pick locations in a subaisle are represented by either two, four, six or eight vertices. The probability $P\left(V_{qi}^{w}=v \,\middle|\, X_{qi}=j\right)$ of v vertices to be required given that j items are to be picked in the subaisle is dependent on the number of storage locations per subaisle and on the way how the items have been assigned to the storage locations in the subaisle. Let f denote the number of storage locations per subaisle. The number of possible assignments of j items to the storage locations then amounts to f^{j}. (Note that some requested items may be assigned to the same location.) Assuming that all assignments have the same probability of being chosen, i.e. items assigned to a subaisle are randomly assigned to a specific storage location in this aisle, $P\left(V_{qi}^{w}=v \,\middle|\, X_{qi}=j\right)$ is approximated via simulation. For $f=50$ and $j \in \{1,\dots,15,76,\dots,90\}$, the results are depicted in Table 1. For each number of requested items j, 10^{8} assignments have been generated randomly and the resulting number of vertices required for the representation has been determined for each assignment. An approximation of $P\left(V_{qi}^{w}=v \,\middle|\, X_{qi}=j\right)$ is then obtained by dividing the number of assignments leading to v vertices by the total number of assignments.

Two vertices are required if the largest gap is defined by a pick location and the adjacent cross aisle (special case 2). With an increasing number of requested items, the number of gaps also increases, which reduces the probability of the gap between a pick location and a cross aisle to be the largest one, leading to a decreasing value of $P\left(V_{qi}^{w}=2 \,\middle|\, X_{qi}=j\right)$. This expectation matches with the results on the left-hand side of the table. However, if the number of requested items to be picked in a subaisle gets very large, $P\left(V_{qi}^{w}=2 \,\middle|\, X_{qi}=j\right)$ increases with an increasing value of j. This can be explained by the fact that lots of gaps exist in this case and several gaps represent a largest gap. Since least vertices are required for the representation according to special case 2, this kind of representation is chosen whenever it is possible. Four vertices are needed if exactly two pick locations exist and the largest gap is situated between these

Table 1: Approximation of $P\left(V_{qi}^{w}=v\,\middle|X_{qi}=j\right)$ [%] for a subaisle including 50 storage locations

	number of vertices required						number of vertices required			
j	**2**	**4**	**6**	**8**		*j*	**2**	**4**	**6**	**8**
1	100.00	0.00	0.00	0.00		76	35.02	0.00	6.05	58.93
2	68.00	32.00	0.00	0.00		77	36.45	0.00	5.89	57.66
3	52.08	3.84	44.09	0.00		78	37.88	0.00	5.73	56.39
4	42.53	0.36	41.62	15.49		79	39.34	0.00	5.58	55.09
5	36.21	0.03	35.91	27.85		80	40.79	0.00	5.42	53.79
6	31.72	0.00	31.44	36.84		81	42.24	0.00	5.27	52.49
7	28.36	0.00	28.08	43.56		82	43.70	0.00	5.12	51.18
8	25.78	0.00	25.47	48.74		83	45.15	0.00	4.97	49.88
9	23.74	0.00	23.40	52.85		84	46.59	0.00	4.83	48.58
10	22.09	0.00	21.73	56.18		85	48.02	0.00	4.69	47.29
11	20.72	0.00	20.34	58.94		86	49.45	0.00	4.54	46.01
12	19.58	0.00	19.18	61.24		87	50.86	0.00	4.40	44.73
13	18.63	0.00	18.20	63.16		88	52.25	0.00	4.27	43.48
14	17.84	0.00	17.38	64.79		89	53.63	0.00	4.14	42.23
15	17.16	0.00	16.66	66.18		90	54.98	0.00	4.01	41.01

locations (special case 3). This case has only to be considered if 2 to 4 requested items are located in the subaisle. For a larger number of items, this case hardly arises because the probability that all items refer to two pick locations gets close to 0. Six vertices are introduced when the largest gap is defined by the two pick locations nearest to a cross aisle (special cases 4 and 5). In general, the probability of this case to appear is fairly even to the probability of special case 2. However, due to the lower number of vertices required, special case 2 is chosen if possible. This implies $P\left(V_{qi}^{w}=2\,\middle|X_{qi}=j\right) \geq P\left(V_{qi}^{w}=6\,\middle|X_{qi}=j\right)$ for all j, where the difference is negligible for small values of j and it gets significant when the number of requested items gets very large. In the standard case of item distribution, eight vertices are required for the representation of the pick locations. Up to a certain point, $P\left(V_{qi}^{w}=8\,\middle|X_{qi}=j\right)$ increases when more requested items are contained in the subaisle, since the number of gaps increases. If the number of gaps gets very large, this probability decreases because of the advantageous alternative "special case 2".

Analogueously, the expected number of arcs $E\left(A^{w}\right)$ representing movements in a subaisle is determined. The probability $P\left(V_{qi}^{w}=a\,\middle|X_{qi}=j\right)$ of a arcs to be required for the representation of j requested items is given by $P\left(V_{qi}^{w}=v\,\middle|X_{qi}=j\right)$ for the corresponding number of vertices v regarding the special cases.

$$E\left(A^{w}\right)=\sum_{q=1}^{p}\sum_{i=1}^{m}\sum_{a\in\{4,8,10,12\}} a\cdot\sum_{j=1}^{n} P\left(V_{qi}^{w}=a\,\middle|X_{qi}=j\right)\cdot\binom{n}{j}\cdot\alpha_{qi}^{j}\cdot\left(1-\alpha_{qi}\right)^{n-j}. \tag{20}$$

Finally, estimations for the total number of vertices $E\left(V\right)$ and arcs $E\left(A\right)$ are given by

$$E\left(V\right)=E\left(V^{c}\right)+E\left(V^{w}\right)\ \text{and} \tag{21}$$

$$E\left(A\right)=E\left(A^{c}\right)+E\left(A^{w}\right). \tag{22}$$

4 Model formulation

Based on the graph defined in the previous section, a mathematical model to the SPRP is formulated. According to Scholz et al. (2016), the model consists of the following components:

- Degree constraints: Each vertex visited has to be left afterwards.
- Subtour elimination constraints: The tour has to be connected.
- Depot inclusion constraint: The depot has to be a part of the tour.
- Pick location inclusion constraints: Each pick location has to be visited for retrieving all items.

Two different types of variables are contained in the model. A binary variable is introduced for each arc of the graph indicating whether the respective arc is included in the tour or not. Using these variables, the degree constraints are formulated by claiming that, for each vertex, its indegree has to be equal to its outdegree. The second type of variables is required for the exclusion of subtours, which is achieved by introducing so-called single-commodity flow constraints (Gavish & Graves, 1978). These constraints are based on the assumption that a single commodity type exists of which one unit has to be delivered to each pick location. For this reason, a non-negative variable is introduced for each arc which describes the amount of the commodity passing the arc. In this way, vertices are enumerated according to their appearance in the tour which guarantees the exclusion of subtours.

Since not all vertices have to be visited, additional constraints are required in order to ensure that the depot is included in the tour and all requested items are retrieved. For guaranteeing the depot to be a part of the tour, the outdegree of the corresponding vertex "0" must not be smaller than 1. The pick location inclusion constraints depend on the respective special case of item distribution of a subaisle i of block q. As for the standard case, no requested items are situated between the pick locations defining the largest gap. Thus, all items of this subaisle are retrieved if either arc $([u,q,i,1],[u,q,i,2])$ or $([d,q,i,3],[u,q,i,4])$ and either arc $([u,q,i,3],[u,q,i,4])$ or $([d,q,i,1],[d,q,i,2])$ are included in the tour. No pick location inclusion constraints are needed for subaisles following special case 1 because no requested items are located in these subaisles. According to special case 2, all pick locations are situated between the locations defined by vertex $[u,q,i,1]$ and $[d,q,i,1]$. The arcs are arranged in such a way that visiting and leaving one of these vertices corresponds to a traversal of the subaisle or to moving to the pick location farthest away from an adjacent cross aisle and then returning to this cross aisle. In both cases, all requested items are retrieved in the subaisle. Thus, it has to be ensured that either vertex $[u,q,i,1]$ or $[d,q,i,1]$ is included in the tour. Subaisles assigned to special case 3 only contain two pick locations. All requested items are collected if both locations are visited. This can be guaranteed if for each pick location at least one vertex of the corresponding vertex pair is contained in the tour. For subaisles belonging to special cases 4 or 5, it is sufficient to ensure that the two locations defining the largest gap are visited. Due to the degree constraints, all requested items are then retrieved.

The model formulation is included in the appendix available at `http://www.mansci.ovgu.de/mansci/en/Research/Materials/2016+_+III_-p-608.html`. Due to the reduction procedures, lots of distinctions of cases are necessary in order to formulate the model, resulting in more than 500 different types of constraints. However, a fraction of the constraints will appear in the model formulation to a specific SPRP instance. In fact, the size of the model linearly increases with the number of vertices and arcs included in the reduced graph, which again linearly increase with the number of subaisles. More precisely, the number of variables equals two times the number of arcs, whereas the number of constraints amounts to two times the number of vertices plus the number of arcs (if the depot and the pick location inclusion constraints are neglected).

5 Numerical experiments

5.1 Setup

Both the model formulation without application of the reduction procedures (referred to as basic model) and the model based on the reduced graph (referred to as reduced model) are now solved by means of a commercial IP-solver in order to evaluate the impact of the reduction procedures on the performance of the solution approach. Extensive numerical experiments are conducted and the model formulations are compared with respect to the size of the formulations, the number of optimal solutions obtained within a given time limit, the optimality gap if no optimal solution has been found as well as the computing time required for the generation of an optimal solution.

The settings for the numerical experiments are adapted from the experiments of Scholz et al. (2016). They considered the SPRP in a single-block layout and demonstrated that their model formulation was able to provide optimal solutions within a small amount of computing time. Due to this good performance, application of procedures for the reduction of the size is not necessary. Therefore, the focus is put on warehouses with 2 and 3 blocks here. As done by Scholz et al. (2016), the number of picking aisles is set to 5, 10, 15, 20, 25 and 30, i.e. the number of subaisles varies between 10 (2 blocks and 5 picking aisles) and 90 (3 blocks and 30 picking aisles). Each subaisle consists of 50 storage locations uniformly arranged on both sides of the subaisle. The length of a single storage location is equal to 1 length unit (LU). The distance between an adjacent cross aisle and the nearest storage location of the subaisle amounts to 1 LU as well. Thus, 26 LUs have to be covered in order to traverse a subaisle. The distance between two adjacent picking aisles equals 5 LUs.

The number of requested items is set to 30, 45, 60, 75 and 90. In each problem instance, the storage locations of the requested items are generated in such a way that the resulting pick locations are mutually different. Furthermore, in order to have a fair comparison, it is ensured that the block farthest from the depot includes at least one pick location. Two different procedures are considered for the assignment of articles to storage locations, namely the random and a class-based storage assignment policy. According to the random assignment strategy, each pick location has the same probability of being included in the tour. When using a class-based storage assignment policy, articles are divided into several classes depending on the demand frequency. As done by Henn & Wäscher (2012), three classes A, B and C are considered, where class A consists of 10% of all articles which possess the highest demand frequency and represent up to 52% of the total demand. 30% of all articles are assigned to class B, where these articles are responsible for 36% of the total demand. The remaining articles are assigned to class C and have quite low demand frequencies. For each class, subaisles are determined based on the distance to the depot. 10% of all subaisles with the shortest distance to the depot are assigned to class A, while 60% of all subaisles farthest away from the depot correspond to class *C*. The remaining subaisles belong to class B. Each article from a class is then randomly assigned to a storage location of a corresponding subaisle.

Combination of the parameters gives rise to 120 different problem classes. For each problem class, 30 instances have been generated, resulting in 3600 instances in total. The basic and the reduced formulations have been solved by CPLEX 12.6.3 on a desktop PC with a 3.4 GHz Pentium processor and 8 GB RAM. The computing time for solving a single instance has been limited to 30 minutes.

5.2 Results

5.2.1 Size of the formulations

The procedures introduced in Section 3 aim for reducing the size of the graph, which results in a reduction of the size of the model formulation as well. In order to evaluate the impact of the procedures on the size of the model, the basic and the reduced model are compared with respect to the number of variables and constraints. In Tables 2 and 3, the number of variables (#var) and the number of constraints (#cons) are depicted for the basic and the reduced model, respectively. Furthermore, the amount of reduction (in %) obtained by the reduction procedures is given. The values in brackets relate to the estimations based on the formulas given in Subsection 3.4. In both tables, problem classes corresponding to large-sized warehouses are considered, i.e. the picking area consists of 3 blocks, while the number of picking aisles *m* is 20, 25 or 30. The number of pick locations *n* is as described in the previous subsection. The size of the formulations is depicted for random (Table 2) and class-based storage assignment (Table 3) strategies.

Table 2: Size of the model formulations in case of a three-block layout and random storage assignment

		basic		**reduced**		**reduction [%]**	
m	*n*	#var	#cons	#var	#cons	#var	#cons
20	30	2812	3435	1468.4 (1480.5)	1743.0 (1685.7)	47.78 (47.35)	49.26 (50.93)
20	45	2812	3435	1630.3 (1607.1)	1934.6 (1851.0)	42.02 (42.85)	43.68 (46.11)
20	60	2812	3435	1725.9 (1700.7)	2041.5 (1979.3)	38.62 (39.52)	40.57 (42.38)
20	75	2812	3435	1808.9 (1777.2)	2144.6 (2088.4)	35.67 (36.80)	37.57 (39.20)
20	90	2812	3435	1874.1 (1842.8)	2221.1 (2185.1)	33.35 (34.47)	35.34 (36.39)
25	30	3532	4295	1820.7 (1809.9)	2154.1 (2043.6)	48.45 (48.76)	49.85 (52.42)
25	45	3532	4295	1975.5 (1955.5)	2336.2 (2230.2)	44.07 (44.63)	45.61 (48.07)
25	60	3532	4295	2079.5 (2062.6)	2451.7 (2373.8)	41.12 (41.60)	42.92 (44.73)
25	75	3532	4295	2168.8 (2150.4)	2552.9 (2495.5)	38.60 (39.12)	40.56 (41.90)
25	90	3532	4295	2261.9 (2225.9)	2660.2 (2603.6)	35.96 (36.98)	38.06 (39.38)
30	30	4252	5155	2137.9 (2136.8)	2487.2 (2398.6)	49.72 (49.75)	51.75 (53.47)
30	45	4252	5155	2307.7 (2299.7)	2696.2 (2604.7)	45.73 (45.91)	47.70 (49.47)
30	60	4252	5155	2442.4 (2418.6)	2858.8 (2761.5)	42.56 (43.12)	44.54 (46.43)
30	75	4252	5155	2538.4 (2516.1)	2970.5 (2894.1)	40.30 (40.83)	42.38 (43.86)
30	90	4252	5155	2632.0 (2600.2)	3080.0 (3011.8)	38.10 (38.85)	40.25 (41.58)

As can be seen, the size of the basic model depends on the number of picking aisles only and is independent of the number of pick locations. In contrast, the size of the reduced model is dependent on both *m* and *n*. It is even dependent on the specific storage locations of the requested items, i.e. each instance may result in a different size. However, compared to the basic formulation, far fewer variables and constraints are required for modeling an instance to the SPRP. For the random assignment policy, the amount of the reduction achieved regarding the number of variables and constraints ranges from 33.35% and 35.34% ($m = 20$, $n = 90$) to 49.72% and 51.75% ($m = 30$, $n = 30$), respectively. The reduction procedures have the largest impact for instances characterized by a large number of picking aisles and a small number of pick locations. In these cases, only a few pick locations have to be visited in a subaisle, which means that lots of vertices and arcs can be removed from the graph by consideration of the special cases of item distribution. Furthermore, if the number of subaisles is large compared to the number of pick locations, many subaisles do not contain any requested items and, thus, the introduction of the pyramid structure leads to considerable size reductions as well.

The estimated amount of reduction ranges from 34.47% and 36.39% ($m = 20$, $n = 90$) to 49.75% and

53.47% ($m = 30$, $n = 30$) regarding the number of variables and constraints, respectively. As it is the case for the average reduction obtained, the estimated amount of reduction increases with an increasing number of picking aisles and a decreasing number of pick locations. The number of variables and constraints is underestimated slightly. This can be explained by the fact that, in the instances from the experiments, the farthest block always includes a requested item and all requested items refer to different pick locations. These adjustments decrease the impact of the reduction procedures and are not considered in the (more general) formulas developed in Subsection 3.4. Nevertheless, very close estimations are obtained by using the formulas. This is particularly true for the number of variables, where the average deviation of the estimation from the average number of variables amounts to 1.03%. For the number of constraints, the average deviation equals 3.15%.

Table 3: Size of the model formulations in case of a three-block layout and class-based storage assignment

		basic		**reduced**		**reduction [%]**	
m	n	#var	#cons	#var	#cons	#var	#cons
20	30	2812	3435	1233.9 (1142.0)	1490.2 (1309.6)	56.12 (59.39)	56.62 (61.87)
20	45	2812	3435	1422.0 (1342.4)	1713.3 (1552.7)	49.43 (52.26)	50.12 (54.80)
20	60	2812	3435	1469.2 (1463.5)	1791.5 (1705.3)	47.75 (47.96)	47.85 (50.36)
20	75	2812	3435	1577.9 (1548.5)	1901.9 (1815.8)	43.89 (44.93)	44.63 (47.14)
20	90	2812	3435	1679.7 (1613.8)	2018.8 (1902.9)	40.27 (42.61)	41.23 (44.60)
25	30	3532	4295	1598.1 (1423.8)	1871.9 (1616.2)	54.75 (59.69)	56.42 (62.37)
25	45	3532	4295	1679.3 (1649.3)	2015.8 (1887.7)	52.45 (53.30)	53.07 (56.05)
25	60	3532	4295	1846.7 (1788.1)	2215.8 (2061.1)	47.72 (49.37)	48.41 (52.01)
25	75	3532	4295	1958.0 (1887.4)	2335.7 (2188.6)	44.56 (46.56)	45.62 (49.04)
25	90	3532	4295	2040.0 (1964.5)	2423.4 (2290.0)	42.24 (44.38)	43.58 (46.68)
30	30	4252	5155	1764.7 (1697.9)	2071.3 (1914.3)	58.50 (60.07)	59.82 (62.87)
30	45	4252	5155	2056.1 (1953.1)	2397.2 (2219.2)	51.64 (54.07)	53.50 (56.95)
30	60	4252	5155	2235.2 (2112.8)	2624.6 (2417.5)	47.43 (50.31)	49.09 (53.10)
30	75	4252	5155	2270.1 (2228.2)	2682.2 (2565.0)	46.61 (47.60)	47.97 (50.24)
30	90	4252	5155	2452.9 (2318.1)	2874.3 (2682.6)	42.31 (45.48)	44.24 (47.96)

When articles are assigned according to the class-based assignment strategy described in Subsection 5.1, the impact of the reduction procedures increases (see Table 3). On average, the amount of reduction ranges from 40.27% and 41.23% ($m = 20$, $n = 90$) to 58.50% and 59.82% ($m = 30$, $n = 30$) for the number of variables and constraints, respectively. Due to the class-based assignment procedure, pick locations near to the depot have a large probability of being included in the tour. Therefore, many subaisles far away from the depot can be removed from the graph by application of the pyramid structure, resulting in a graph which is much smaller than in case of random assignment.

The quality of the estimations deteriorates if the instances based on the class-based assignment strategy are considered. This can also be explained by the instance generation procedure. In case of random assignment, articles are distributed over a large number of subaisles. It is very likely that all blocks contain requested items and that all requested items refer to different pick locations since few items are included in the same subaisle. Therefore, minor adjustments have to be performed in the instance generation process only. This is not true in case of class-based assignment. A large proportion of the requested items will be located in the subaisles near to the depot. In particular, if the number of items is quite small, the probability of the block farthest from the depot not containing a pick location is not negligible at all. Thus, it is not surprising that the average deviation of the estimated from the average number of variables and constraints increases, and now amounts to 4.27% and 7.29%, respectively. Nevertheless, the estimations still represent a good approximation of the size of the reduced model and can be used in order to evaluate in advance whether application of the reduction procedures to an instance of a certain problem class is a worthwhile endeavor.

5.2.2 Number of optimal solutions and optimality gaps

Since the reduced formulation includes considerable fewer variables and constraints, while having the same components as the basic model, it can be expected that application of an IP-solver using this formulation leads to a better performing solution approach to the SPRP. This expectation is verified by the results of the numerical experiments. In Table 4, the number of instances solved to optimality within 30 minutes of computing time is depicted for both formulations. If a number is equal to 30, all instances of the problem class have been solved optimally by using the respective formulation.

Concerning the basic model, it can be observed that its performance is independent of the number of pick locations, which can be explained by the fact that each subaisle is represented by a constant number of vertices in the underlying graph. The same line of argumentation holds for the storage assignment procedure which affects the locations of the requested items. Two parameters can be identified which have an impact on the number of optimal solutions obtained within the time limit. First, an increasing number of picking aisles results in a decreasing number of optimal solutions. Using the basic formulation, all instances of problem classes with 5 or 10 picking aisles can be solved to proven optimality. However, this number rapidly decreases when the warehouse includes more picking aisles. For problem classes with 20 picking aisles, an optimal solution can be determined for approximately half of all instances, whereas only 4 out of 120 instances with 30 picking aisles have been solved to optimality. This performance matches with the observations of Scholz et al. (2016). The second important parameter is the number of blocks. An increasing number of blocks leads to fewer optimal solutions obtained. For 2 blocks, nearly all instances with 15 picking aisles or fewer can be solved to optimality. However, for 3 blocks, only approximately two-thirds of the instances with 15 picking

Table 4: Number of optimally solved instances (out of 30) within 30 minutes of computing time

		random storage assignment				class-based storage assignment			
		2 blocks		3 blocks		2 blocks		3 blocks	
m	n	**basic**	**reduced**	**basic**	**reduced**	**basic**	**reduced**	**basic**	**reduced**
5	30	30	30	30	30	30	30	30	30
5	45	30	30	30	30	30	30	30	30
5	60	30	30	30	30	30	30	30	30
5	75	30	30	30	30	30	30	30	30
5	90	30	30	30	30	30	30	30	30
10	30	30	30	30	30	30	30	30	30
10	45	30	30	30	30	30	30	30	30
10	60	30	30	30	30	30	30	30	30
10	75	30	30	30	30	30	30	30	30
10	90	30	30	30	30	30	30	30	30
15	30	30	30	29	30	27	30	22	30
15	45	30	30	29	30	29	30	21	30
15	60	30	30	28	30	29	30	26	30
15	75	29	30	28	30	30	30	20	30
15	90	27	30	27	30	27	30	19	30
20	30	20	30	7	30	17	30	8	30
20	45	18	30	6	30	16	30	10	30
20	60	19	29	10	30	15	30	6	30
20	75	15	29	9	30	17	30	11	29
20	90	10	29	8	29	17	30	11	29
25	30	8	30	0	30	2	30	0	30
25	45	4	30	0	30	2	30	0	30
25	60	3	30	0	30	7	30	0	30
25	75	6	29	1	30	5	30	1	30
25	90	6	23	2	29	8	28	0	30
30	30	0	30	0	30	0	30	0	30
30	45	2	30	0	30	1	29	0	30
30	60	1	30	0	30	0	27	0	29
30	75	0	26	0	28	0	25	0	29
30	90	0	22	0	29	0	20	0	27

aisles can be solved optimally. Furthermore, an optimal solution has been found for 4 instances with $m \geq 25$ only. This performance was also expected because the size of the formulation is multiplied by the number of blocks. The results of the experiments indicate that the basic formulation is not suitable for solving SPRPs in multi-block layouts.

When using the reduced model, most of the instances can be solved to optimality. For all instances with 15 picking aisles or fewer, optimal solutions have been found. If the number of picking aisles is increased to 20 or 25, except for one problem class, at least 28 out of 30 instances can still be solved, respectively. If m gets very large ($m = 30$), several instances exist to which no optimal solution has been determined. However, the minimum number of optimal solutions obtained in a problem class amounts to 20 (2 blocks, 30 picking aisles, 90 pick locations, class-based storage assignment), which shows that application of the reduced model outperforms the usage of the basic formulation by far. Besides the number of picking aisles, an increasing number of pick locations has a negative impact on the performance of the reduced model. This is not surprising since fewer vertices and arcs can be removed from the underlying graph, if more pick locations have to be represented, leading to a deteriorating performance of the approach. The storage assignment procedure does not seem to have an impact on the number of optimal solutions obtained. In case of the class-based assignment policy, lots of subaisles can be removed from the graph by introduction of the pyramid structure, resulting in a significant reduction of the size of the model (see Table 3). This is not possible when articles are assigned according to the random assignment policy. However, in case of the random assignment, the number of pick locations per subaisle is quite low, which allows for large reductions based on the special cases of item distribution (see Table 1). Thus, it can be observed that the interaction of the reduction procedures leads to a good performance of the reduced model for both storage assignment policies. The major advantage of the reduced model can be seen in its performance for an increasing number of blocks. Compared to the two-block case, the number of optimal solutions obtained even increases when 3 blocks are considered. In fact, at least 90% (27 out of 30) of the instances of a problem class have been solved to optimality. This clearly shows that the reduced model is able to deal with multiple blocks.

Besides the number of optimal solutions obtained, the quality of solutions is investigated if no optimal solution has been found, i.e. if the solution process has been terminated after 30 minutes of computing time. In Table 5, the maximum optimality gaps are depicted for problem classes with a larger number of picking aisles ($m \geq 20$). If the maximum gap amounts to 0.00%, all instances of the corresponding class have been solved to optimality.

Table 5: Maximum optimality gaps [%]

		random storage assignment				class-based storage assignment			
		2 blocks		3 blocks		2 blocks		3 blocks	
m	*n*	**basic**	**reduced**	**basic**	**reduced**	**basic**	**reduced**	**basic**	**reduced**
20	30	3.47	0.00	8.73	0.00	7.28	0.00	8.10	0.00
20	45	2.43	0.00	3.39	0.00	3.98	0.00	7.54	0.00
20	60	2.59	1.01	1.91	0.00	4.66	0.00	6.03	0.00
20	75	3.50	0.70	2.42	0.00	3.36	0.00	3.74	0.62
20	90	2.45	0.56	2.53	0.43	5.24	0.00	5.63	1.05
25	30	7.94	0.00	11.31	0.00	8.62	0.00	12.99	0.00
25	45	5.68	0.00	5.70	0.00	7.98	0.00	8.89	0.00
25	60	4.56	0.00	5.10	0.00	9.55	0.00	10.40	0.00
25	75	4.61	1.35	4.20	0.00	5.71	0.00	6.80	0.00
25	90	3.64	1.40	3.38	0.30	4.79	1.02	7.08	0.00
30	30	12.11	0.00	15.29	0.00	16.41	0.00	20.62	0.00
30	45	7.91	0.00	8.93	0.00	10.60	0.20	16.04	0.00
30	60	6.23	0.00	7.30	0.00	8.89	4.11	10.05	0.43
30	75	5.35	1.33	6.09	0.79	7.36	1.63	8.34	1.37
30	90	2.87	1.19	4.63	0.20	8.26	4.50	7.84	2.83

As expected, for the basic formulation, the maximum optimality gap increases with an increasing number of picking aisles as well as an increasing number of blocks. In contrast, the gaps decrease with an increasing number of pick locations and when articles are randomly assigned to storage locations. This can be explained by the fact that the minimum tour length increases in these cases. Thus, small changes to the tour do not have such a large impact on the relative deviation from the optimal tour length. Nevertheless, it can be observed that application of the basic formulation leads to tours which are up to 20.62% longer than an optimal tour (3 blocks, 30 picking aisles, 30 pick locations, class-based assignment) and, therefore, it is again concluded that this formulation is not suitable for dealing with SPRPs including multiple blocks.

The maximum optimality gaps obtained by application of the reduced model are much smaller. As it is the case for the number of optimal solutions found, the performance deteriorates with an increasing number of picking aisles and pick locations, whereas it slightly improves with an increasing number of blocks. Furthermore, it can be seen that the gaps are smaller when the random assignment procedure is used. In this case, the maximum gap amounts to 1.40% (2 blocks, 25 picking aisles, 90 pick locations), which shows that at least near-optimal solutions have been found by application of the reduced model. If the class-based storage assignment procedure is assumed, the gaps are getting larger for instances with 30 picking aisles. However, all gaps are not larger than 4.50%, which means that near-optimal solutions have also been provided in case of the class-based assignment.

5.2.3 Computing times

The average computing time required for the determination of an optimal solution by application of the respective model formulation is depicted in Table 6. If no optimal solution has been found within the predefined time interval, a computing time of 30 minutes has been reported. An average computing time of 1800.00 seconds means that no instance of the problem class has been solved to proven optimality.

The number of picking aisles and the number of blocks have an impact on the computing time required for applying the basic formulation, whereas no conclusions can be drawn regarding the impact of the number of pick locations and the storage assignment strategy. Computing times rapidly increase with an increasing number of picking aisles. While instances with up to 10 picking aisles can be solved within a few seconds on average, up to 9 minutes are required for $m = 15$ and even up to 28 minutes when 20 picking aisles are considered. For instances with more than 20 picking aisles, computing times are not meaningful because not many optimal solutions have been found in this case. Concerning the number of blocks, it can be observed that computing times increase significantly if three blocks have to

Table 6: Computing times [sec]

		random storage assignment				class-based storage assignment			
		2 blocks		3 blocks		2 blocks		3 blocks	
m	n	**basic**	**reduced**	**basic**	**reduced**	**basic**	**reduced**	**basic**	**reduced**
5	30	0.48	0.31	0.95	0.40	0.78	0.44	0.99	0.37
5	45	0.43	0.32	0.88	0.47	0.71	0.47	0.98	0.40
5	60	0.49	0.39	0.84	0.52	0.67	0.46	0.84	0.44
5	75	0.49	0.39	0.80	0.60	0.74	0.52	1.14	0.47
5	90	0.53	0.42	1.00	0.75	0.78	0.55	1.21	0.56
10	30	5.87	1.10	14.98	1.86	14.29	1.03	20.71	1.57
10	45	8.92	2.00	34.70	3.23	12.17	1.42	35.52	1.83
10	60	10.61	3.61	17.57	3.93	13.69	1.38	21.11	1.79
10	75	14.98	8.68	29.88	6.80	10.85	1.58	30.20	2.31
10	90	21.91	11.05	58.24	11.18	10.03	1.42	29.93	3.16
15	30	92.56	3.96	249.44	4.74	428.07	6.08	525.43	14.00
15	45	66.11	5.43	315.19	6.73	351.52	6.54	428.78	6.44
15	60	149.77	18.72	273.87	12.14	355.98	19.50	397.78	7.20
15	75	316.32	50.73	304.36	18.33	271.85	13.10	593.61	17.79
15	90	524.69	67.56	332.93	34.29	478.34	47.94	525.43	14.00
20	30	1028.47	9.40	1584.00	8.66	1000.09	6.91	1390.47	7.25
20	45	974.57	17.19	1582.97	14.18	1010.82	16.44	1606.71	17.06
20	60	911.62	91.53	1279.95	18.13	1110.61	53.12	1704.86	66.37
20	75	1074.15	128.81	1506.79	41.43	1111.41	113.47	1309.55	100.81
20	90	1393.30	258.85	1602.78	149.07	1015.24	168.85	1277.25	108.43
25	30	1572.69	18.68	1802.00	18.17	1695.72	20.02	1800.00	18.47
25	45	1712.87	33.48	1800.00	21.05	1727.68	44.54	1800.00	22.50
25	60	1743.07	77.50	1800.00	49.85	1517.52	162.40	1800.00	93.28
25	75	1592.18	229.65	1756.90	70.03	1654.45	270.24	1766.42	170.96
25	90	1620.58	664.22	1726.38	251.04	1529.55	317.64	1800.00	205.47
30	30	1800.41	46.39	1800.00	61.85	1801.15	59.00	1800.00	71.38
30	45	1703.87	74.41	1800.00	96.15	1752.28	184.08	1800.00	58.65
30	60	1773.87	167.90	1800.00	82.96	1800.90	330.98	1800.50	232.28
30	75	1800.00	515.90	1800.00	227.76	1801.73	581.39	1800.47	155.85
30	90	1800.00	771.55	1800.00	306.88	1802.37	904.34	1800.00	339.70

be dealt with. In fact, for instances with $m \leq 20$, computing times rise by 81.31% on average if three instead of two blocks are considered.

For the reduced formulation, the results depicted in Table 6 also match with the expectations based on the development of the size of the formulation (see Tables 2 and 3). Both the number of picking aisles and the number of pick locations have an impact on its performance. Regarding the number of picking aisles m, all instances with $m \leq 15$ have been solved within less than 70 seconds on average, whereas up to 15 minutes are required for solving instances with $m = 30$. When m gets large, the impact of the number of pick locations n gets significant as well. While the maximum average computing time amounts to 1.2 minutes (3 blocks, 30 picking aisles, class-based assignment) for instances with a small number of pick locations ($n = 30$), on average up to 15 minutes are required for solving instances with the same characteristics but 90 pick locations. Concerning the storage assignment policy, it can be seen that computing times are lower for class-based assignment if instances with $m \leq 20$ are considered. Otherwise, the performance of the reduced model seems to be better for random assignment. As observed before, the performance of the reduced model improves with an increasing number of blocks. This is particularly true for instances difficult to solve, i.e. instances with a large number of picking aisles or with many pick locations. The largest reduction can be seen in the problem classes with 30 picking aisles, 75 pick locations and class-based storage assignment, where an increase of the number of blocks leads to a decrease of the average computing time by 73%.

Compared to the basic formulation, application of the reduced model consumes far less computing time. It can be observed that the impact of the number of picking aisles on the computing time is much smaller for the reduced formulation. This particularly holds if a small number of pick locations ($n = 30$) is considered since lots of vertices can be removed from the underlying graph. Application of the reduction procedures results in a decrease of the average computing time by up to 99.5% (3 blocks, 20 picking aisles, class-based assignment). When more pick locations are considered, the computing time required for solving the reduced model increases. However, even for a very large number of pick locations ($n = 90$), reductions of more than 90% (3 blocks, 20 picking aisles) are achieved for both random and class-based storage assignment. Furthermore, it can be observed that the largest amount of reduction is obtained in case of three blocks. Whereas computing times significantly increase for the basic formulation, computing times even decrease for the reduced model. This performance shows that application of the reduction procedures is pivotal in order to obtain a problem-specific model formulation able to deal with SPRPs in warehouses with multiple blocks.

6 Conclusion and Outlook

The Single-Picker Routing Problem deals with the determination of the sequence according to which requested items are to be retrieved from the storage locations in the picking area of the warehouse. It represents a special case of the Traveling Salesman Problem. However, Scholz et al. (2016) pointed out that problem-specific solution approaches lead to better results and can easier be adapted to modifications arising in practical applications. They proposed a mathematical programming formulation to the Single-Picker Routing Problem in a single-block layout, which is extended to the case of multiple blocks in this paper. A so-called pyramid structure is introduced and special cases of item distribution are identified, which significantly reduce the size of the formulation and make the formulation suitable for dealing with large instances. Since the size of the resulting reduced model is not dependent on the size of the problem only but it also depends on the specific locations of the requested items, the impact of the reduction procedures on the size of the model cannot be determined in advance. Therefore, formulas are developed, which provide good estimations for the number of variables and constraints and allow for evaluating whether application of the reduction procedures to an instance is a worthwhile endeavor.

By means of numerical experiments, the proposed reduced model formulation is evaluated and compared to the basic formulation to which no reduction procedures have been applied. It is shown that the reduced model formulation outperforms the basic formulation by far in terms of optimal solutions found, optimality gaps and computing times. By application of a commercial IP-solver to the reduced model, nearly all instances from the experiments have been solved to optimality within the given time limit, whereas the basic model was able to solve instances with a small number of picking aisles only. Regarding average computing times, reductions of up to 99.5% have been obtained, which demonstrates that application of the reduction procedures is inevitable when dealing with multi-block layouts. Finally, it is observed that computing times do not increase with an increasing number of blocks, which is a major advantage of the reduced model since no efficient solution approach exists so far, which is able to deal with more than two blocks.

The model formulation has been designed for picking areas following a block layout. However, a recent trend is to design the layout of a warehouse without using parallel picking and cross aisles. Instead, non-conventional layouts such as fishbone layouts are applied (Çelik & Süral, 2014). The construction of problem-specific formulations to those layouts would be a promising area for future research.

Further research could also concentrate on picker blocking aspects. In this paper, it is assumed that aisles are wide enough enabling order pickers to pass each other. Without losing generality, it can then be assumed that only one picker is available when dealing with the routing problem because routes can be determined independently of each other. This is not true if very narrow aisles have to be dealt with. Thus, it would be interesting to investigate whether blocking aspects could be integrated into the model.

References

Burkard, R.; Deneko, V. G; van der Veen, J. A. A. & Woeginger, G. J. (1998): Well-Solvable Special Cases of the Traveling Salesman Problem: A Survey. *SIAM Review* 40, 496-546.

Caron, F.; Marchet, G. & Perego, A. (2000): Optimal Layout in Low-Level Picker-to-Part Systems. *International Journal of Production Research* 38, 101-117.

Çelik, M.; Süral, H. (2014): Order Picking under Random and Turnover-Based Storage Policies in Fishbone Aisle Warehouses. *IIE Transactions* 46, 283-300.

Dantzig. G. B.; Fulkerson, D. R. & Johnson S. M. (1954): Solution of a Large-Scale Traveling Salesman Problem. *Journal of the Operations Research Society of America* 2, 363-410.

de Koster, R. & van der Poort, E. (1998): Routing Orderpickers in a Warehouse: A Comparison between Optimal and Heuristic Solutions. *IIE Transactions* 30, 469-480.

de Koster, R.; Le-Duc, T. & Roodbergen, K. J. (2007): Design and Control of Warehouse Order Picking: A Literature Review. *Science Direct* 182, 481-501.

Gavish, B. & Graves, S. C. (1978): The Traveling Salesman Problem and Related Problems. Working Paper GR-078-78, Operations Research Center, Massachusetts Institute of Technology.

Gouveia, L. & Pires, J. M. (2001): The Asymmetric Travelling Salesman Problem: On Generalizations of Disaggregated Miller-Tucker-Zemlin Constraints. *Discrete Applied Mathematics* 112, 129-145.

Henn, S. & Wäscher, G. (2012): Tabu Search Heuristics for the Order Batching Problem in Manual Order Picking Systems. *European Journal of Operational Research* 222, 484-494.

Jarvis, J. M. & McDowell, E. D. (1991): Optimal Product Layout in an Order Picking Warehouse. *IIE Transactions* 23, 93-102.

Letchford, A. N.; Nasiri, S. D. & Theis, D. O. (2013): Compact Formulations of the Steiner Traveling Salesman Problem and Related Problems. *European Journal of Operational Research* 228, 83-92.

Öncan, T.; Altinel, K. & Laporte, G. (2009): A Comparative Analysis of Several Asymmetric Traveling Salesman Problem Formulations. *Computers & Operations Research* 36, 637-654.

Padberg, M.; & Sung, T. (1991): An Analytical Comparison of Different Formulations of the Traveling Salesman Problem. *Mathematical Programming* 52, 315-357.

Petersen, C. G. & Schmenner, R. W. (1999): An Evaluation of Routing and Volume-Based Storage Policies in an Order Picking Operation. *Decision Science* 30, 481-501.

Ratliff, H. D. & Rosenthal, A. R. (1983): Order-Picking in a Rectangular Warehouse: A Solvable Case of the Traveling Salesman Problem. *Operations Research* 31, 507-521.

Roodbergen, K. J. (2001): Layout and Routing Methods for Warehouses. Trial: Rotterdam.

Roodbergen, K. J. & de Koster, R. (2001): Routing Order Pickers in a Warehouse with a Middle Aisle. *European Journal of Operational Research* 133, 32-43.

Scholz, A.; Henn, S.; Stuhlmann, M. & Wäscher, G. (2016): A New Mathematical Programming Formulation for the Single-Picker Routing Problem. *European Journal of Operational Research* 253, 68-84.

Tompkins, J. A.; White, J. A.; Bozer, Y. A. & Tanchoco, J. M. A. (2010): Facilities Planning. 4th edition, John Wiley & Sons: New Jersey.

Wäscher, G. (2004): Order Picking: A Survey of Planning Problems and Methods. *Supply Chain Management and Reverse Logistics*, Dyckhoff , H.; Lackes, R. & Reese, J. (eds.), 323-347, Springer: Berlin.

Appendix: Model formulation to the SPRP

Sets:

$B = \{1,\ldots,p\}$:	set of blocks
$F = \{1,\ldots,p+1\}$:	set of cross aisles
$I^q = \{\underline{m}_q, \overline{m}_q\}$:	set of subaisles to be considered in block q, with $\underline{m}_1 = 1$ and $\underline{m}_q < \overline{m}_q - 1 \;\forall\, q \in B$ (see Section 4.1 for the determination of $\underline{m}_q$ and $\overline{m}_q$)
$I_0^q \subseteq I^q$:	set of the subaisles which are located in block q and follow the standard case of item distribution; $I_0^q = I^q \backslash \{I_1^q \cup I_2^q \cup I_3^q \cup I_4^q \cup I_5^q\}$
$I_1^q \subseteq I^q$:	set of the subaisles which are located in block q and do not contain pick locations (special case 1)
$I_2^q \subseteq I^q$:	set of the subaisles which are located in block q and contain the largest gap between the first or the last pick location and the adjacent cross aisle (special case 2)
$I_3^q \subseteq I^q$:	set of the subaisles which are located in block q and contain exactly two pick locations with the largest gap located between them (special case 3)
$I_4^q \subseteq I^q$:	set of the subaisles which are located in block q and contain the largest gap between the first and the adjacent pick location (special case 4)
$I_5^q \subseteq I^q$:	set of the subaisles which are located in block q and contain the largest gap between the last and the adjacent pick location (special case 5)

Binary variables indicating the arcs included in the tour:

r_{qi}^r : binary variable, $\forall\, (q,i) \in F \times (I^q \backslash \{\overline{m}_q - 1, \overline{m}_q\})$, with

$$r_{qi}^r = \begin{cases} 1, & \text{if arc } ([r,q,i],[r,q,i+1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

r_{qi}^u : binary variable, $\forall\, (q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash \{\overline{m}_q\})$, with

$$r_{qi}^u = \begin{cases} 1, & \text{if arc } ([r,q,i],[u,q,i+1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

r_{qi}^d : binary variable, $\forall\, (q,i) \in (F \backslash \{1\}) \times (I^q \backslash \{\overline{m}_q\})$, with

$$r_{qi}^d = \begin{cases} 1, & \text{if arc } ([r,q,i],[d,q,i+1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

l_{qi}^l : binary variable, $\forall\, (q,i) \in F \times ((I^q \backslash \{\underline{m}_q, \underline{m}_q + 1\}) \cup \{(1,2)\})$, with

$$l_{qi}^l = \begin{cases} 1, & \text{if arc } ([l,q,i],[l,q,i-1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

l^u_{qi} : binary variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash \{\underline{m}_q\})$, with

$$l^u_{qi} = \begin{cases} 1, & \text{if arc } ([l,q,i],[u,q,i-1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

l^d_{qi} : binary variable, $\forall\ (q,i) \in (F \backslash \{1\}) \times (I^q \backslash \{\underline{m}_q\})$, with

$$l^d_{qi} = \begin{cases} 1, & \text{if arc } ([l,q,i],[d,q,i-1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

e^u_{qi1} : binary variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1)$, with

$$e^u_{qi1} = \begin{cases} 1, & \text{if arc } ([d,q+1,i],[u,q,i,1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

e^d_{qi1} : binary variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1)$, with

$$e^d_{qi1} = \begin{cases} 1, & \text{if arc } ([u,q,i],[d,q,i,1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

e^u_{qi2} : binary variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q_3 \cup I^q_4)$, with

$$e^u_{qi2} = \begin{cases} 1, & \text{if arc } ([d,q+1,i],[u,q,i,2]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

e^d_{qi2} : binary variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q_3 \cup I^q_4)$, with

$$e^d_{qi2} = \begin{cases} 1, & \text{if arc } ([u,q,i],[d,q,i,2]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

e^u_{qi3} : binary variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q_0 \cup I^q_5)$, with

$$e^u_{qi3} = \begin{cases} 1, & \text{if arc } ([d,q+1,i],[u,q,i,3]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

e^d_{qi3} : binary variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q_0 \cup I^q_4)$, with

$$e^d_{qi3} = \begin{cases} 1, & \text{if arc } ([u,q,i],[d,q,i,3]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

t_{qi}^{u} : binary variable, $\forall\,(q,i) \in (F\backslash\{p+1\}) \times (I^q\backslash I_1^q)$, with

$$t_{qi}^{u} = \begin{cases} 1, & \text{if arc } ([u,q,i],[u,q,i,1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

t_{qi}^{d} : binary variable, $\forall\,(q,i) \in (F\backslash\{p+1\}) \times (I^q\backslash I_1^q)$, with

$$t_{qi}^{d} = \begin{cases} 1, & \text{if arc } ([d,q+1,i],[d,q,i,1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

w_{qis}^{u} : binary variable, $\forall\,(q,i,s) \in B \times ((I_3^q \times \{1\}) \cup ((I_4^q \cup I_5^q) \times \{1,2\}) \cup (I_0^q \times \{1,2,3\}))$, with

$$w_{qis}^{u} = \begin{cases} 1, & \text{if arc } ([u,q,i,s],[u,q,i,s+1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

w_{qis}^{d} : binary variable, $\forall\,(q,i,s) \in B \times ((I_3^q \times \{1\}) \cup ((I_4^q \cup I_5^q) \times \{1,2\}) \cup (I_0^q \times \{1,2,3\}))$, with

$$w_{qis}^{d} = \begin{cases} 1, & \text{if arc } ([d,q,i,s],[d,q,i,s+1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

v_{qi}^{u} : binary variable, $\forall\,(q,i) \in (B\backslash\{p\}) \times I^q$, with

$$v_{qi}^{u} = \begin{cases} 1, & \text{if arc } ([u,q,i],[u,q+1,i]) \text{ for } i \in I_1^q,\ ([u,q,i,1],[u,q+1,i]) \text{ for } i \in I_2^q, \\ & ([u,q,i,2],[u,q+1,i]) \text{ for } i \in I_3^q,\ ([u,q,i,3],[u,q+1,i]) \text{ for } i \in I_4^q \cup I_5^q \text{ or} \\ & ([u,q,i,4],[u,q+1,i]) \text{ for } i \in I_0^q \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

$v_{qi}^{u,l}$: binary variable, $\forall\,(q,i) \in B \times (I^q\backslash\{\underline{m}_q\})$, with

$$v_{qi}^{u,l} = \begin{cases} 1, & \text{if arc } ([u,q,i],[l,q+1,i]) \text{ for } i \in I_1^q,\ ([u,q,i,1],[l,q+1,i]) \text{ for } i \in I_2^q, \\ & ([u,q,i,2],[l,q+1,i]) \text{ for } i \in I_3^q,\ ([u,q,i,3],[l,q+1,i]) \text{ for } i \in I_4^q \cup I_5^q \text{ or} \\ & ([u,q,i,4],[l,q+1,i]) \text{ for } i \in I_0^q \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

$v^{u,r}_{qi}$: binary variable, $\forall\ (q,i) \in B \times (I^q \backslash \{\overline{m}_q\})$, with

$$v^{u,r}_{qi} = \begin{cases} 1, & \text{if arc } ([u,q,i],[r,q+1,i]) \text{ for } i \in I^q_1,\ ([u,q,i,1],[r,q+1,i]) \text{ for } i \in I^q_2, \\ & ([u,q,i,2],[r,q+1,i]) \text{ for } i \in I^q_3,\ ([u,q,i,3],[r,q+1,i]) \text{ for } i \in I^q_4 \cup I^q_5 \text{ or} \\ & ([u,q,i,4],[r,q+1,i]) \text{ for } i \in I^q_0 \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

v^{d}_{qi} : binary variable, $\forall\ (q,i) \in (B\backslash\{1\}) \times I^q$, with

$$v^{d}_{qi} = \begin{cases} 1, & \text{if arc } ([d,q+1,i],[d,q,i]) \text{ for } i \in I^q_1,\ ([d,q,i,1],[d,q,i]) \text{ for } i \in I^q_2, \\ & ([d,q,i,2],[d,q,i]) \text{ for } i \in I^q_3,\ ([d,q,i,3],[d,q,i]) \text{ for } i \in I^q_4 \cup I^q_5 \text{ or} \\ & ([d,q,i,4],[d,q,i]) \text{ for } i \in I^q_0 \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

$v^{d,l}_{qi}$: binary variable, $\forall\ (q,i) \in B \times (I^q \backslash \{\underline{m}_q\}) \cup \{(1,1)\})$, with

$$v^{d,l}_{qi} = \begin{cases} 1, & \text{if arc } ([u,q+1,i],[l,q,i]) \text{ for } i \in I^q_1,\ ([d,q,i,1],[l,q,i]) \text{ for } i \in I^q_2, \\ & ([d,q,i,2],[l,q,i]) \text{ for } i \in I^q_3,\ ([d,q,i,3],[l,q,i]) \text{ for } \in I^q_4 \cup I^q_5 \text{ or} \\ & ([d,q,i,4],[l,q,i]) \text{ for } i \in I^q_0 \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

$v^{d,r}_{qi}$: binary variable, $\forall\ (q,i) \in B \times (I^q \backslash \{\overline{m}_q\})$, with

$$v^{d,r}_{qi} = \begin{cases} 1, & \text{if arc } ([u,q+1,i],[r,q,i]) \text{ for } i \in I^q_1,\ ([u,q,i,1],[r,q,i]) \text{ for } i \in I^q_2, \\ & ([u,q,i,2],[r,q,i]) \text{ for } i \in I^q_3,\ ([u,q,i,3],[r,q,i]) \text{ for } i \in I^q_4 \cup I^q_5 \text{ or} \\ & ([u,q,i,4],[r,q,i]) \text{ for } i \in I^q_0 \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

y^0_α : binary variable, $\forall\ \alpha \in \{l,r,u\}$, with

$$y^0_\alpha = \begin{cases} 1, & \text{if arc } ([0],[\alpha,1,1]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

y^l_0 : binary variable, with

$$y^l_0 = \begin{cases} 1, & \text{if arc } ([l,1,1],[0]) \text{ is contained in the tour} \\ 0, & \text{otherwise} \end{cases}$$

Real-valued variables to exclude subtours:

$\tilde{r}^r_{qi}$: real-valued variable, $\forall\ (q,i) \in F \times (I^q \backslash \{\overline{m}_q - 1, \overline{m}_q\})$

$\tilde{r}^u_{qi}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash \{\overline{m}_q\})$

$\tilde{r}^d_{qi}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{1\}) \times (I^q \backslash \{\overline{m}_q\})$

$\tilde{l}^l_{qi}$: real-valued variable, $\forall\ (q,i) \in F \times ((I^q \backslash \{\underline{m}_q, \underline{m}_q + 1\}) \cup \{(1,2)\})$

$\tilde{l}^u_{qi}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash \{\underline{m}_q\})$

$\tilde{l}^d_{qi}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{1\}) \times (I^q \backslash \{\underline{m}_q\})$

$\tilde{e}^u_{qi1}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1)$

$\tilde{e}^d_{qi1}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1)$

$\tilde{e}^u_{qi2}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q_3 \cup I^q_4)$

$\tilde{e}^d_{qi2}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q_3 \cup I^q_5)$

$\tilde{e}^u_{qi3}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q_0 \cup I^q_5)$

$\tilde{e}^d_{qi3}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q_0 \cup I^q_4)$

$\tilde{t}^u_{qi}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1)$

$\tilde{t}^d_{qi}$: real-valued variable, $\forall\ (q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1)$

$\tilde{w}^u_{qis}$: real-valued variable, $\forall\ (q,i,s) \in B \times ((I^q_3 \times \{1\}) \cup ((I^q_4 \cup I^q_5) \times \{1,2\}) \cup (I^q_0 \times \{1,2,3\}))$

$\tilde{w}^d_{qis}$: real-valued variable, $\forall\ (q,i,s) \in B \times ((I^q_3 \times \{1\}) \cup ((I^q_4 \cup I^q_5) \times \{1,2\}) \cup (I^q_0 \times \{1,2,3\}))$

$\tilde{v}^u_{qi}$: real-valued variable, $\forall\ (q,i) \in (B \backslash \{p\}) \times I^q$

$\tilde{v}^{u,l}_{qi}$: real-valued variable, $\forall\ (q,i) \in B \times (I^q \backslash \{\underline{m}_q\})$

$\tilde{v}^{u,r}_{qi}$: real-valued variable, $\forall\ (q,i) \in B \times (I^q \backslash \{\overline{m}_q\})$

$\tilde{v}^d_{qi}$: real-valued variable, $\forall\ (q,i) \in (B \backslash \{1\}) \times I^q$

$\tilde{v}^{d,l}_{qi}$: real-valued variable, $\forall\ (q,i) \in B \times (I^q \backslash \{\underline{m}_q\}) \cup \{(1,1)\})$

$\tilde{v}^{d,r}_{qi}$: real-valued variable, $\forall\ (q,i) \in B \times (I^q \backslash \{\overline{m}_q\})$

$\tilde{y}^0_\alpha$: real-valued variable, $\forall\ \alpha \in \{l,r,u\}$

$\tilde{y}^l_0$: real-valued variable

Constants:

c_0 : distance between the depot and the intersection of cross aisle 1 with the first sub-aisle of block 1

c^a : distance between two adjacent picking aisles

$\overline{c}$: length of a sub-aisle

$c^{t,u}_{qi}$: distance between cross aisle q and vertex $[u,q,i,1]$, $\forall\ (q,i) \in B \times (I^q \backslash I^q_1)$

$c^{t,d}_{qi}$: distance between cross aisle $q+1$ and vertex $[d,q,i,1]$, $\forall\ (q,i) \in B \times (I^q \backslash I^q_1)$

$c^{e,u}_{qis}$: distance between cross aisle $q+1$ and vertex $[u,q,i,s]$, $\forall\ (q,i,s) \in B \times (((I^q \backslash I^q_1) \times \{1\}) \cup ((I^q_3 \cup I^q_4) \times \{2\}) \cup ((I^q_0 \cup I^q_5) \times \{3\}))$

$c^{e,d}_{qis}$: distance between cross aisle q and vertex $[d,q,i,s]$, $\forall\ (q,i,s) \in B \times (((I^q \backslash I^q_1) \times \{1\}) \cup ((I^q_3 \cup I^q_5) \times \{2\}) \cup ((I^q_0 \cup I^q_4) \times \{3\}))$

$c^{w,u}_{qis}$: distance between vertex $[u,q,i,s]$ and vertex $[u,q,i,s+1]$, $\forall\ (q,i,s) \in B \times ((I^q_3 \times \{1\}) \cup ((I^q_4 \cup I^q_5) \times \{1,2\}) \cup (I^q_0 \times \{1,2,3\}))$

$c^{w,d}_{qis}$: distance between vertex $[d,q,i,s]$ and vertex $[d,q,i,s+1]$, $\forall\ (q,i,s) \in B \times ((I^q_3 \times \{1\}) \cup ((I^q_4 \cup I^q_5) \times \{1,2\}) \cup (I^q_0 \times \{1,2,3\}))$

M : large number (e.g. number of vertices)

Objective Function:

$$
\begin{aligned}
\min\ & \sum_{q=1}^{p+1} \sum_{i=\underline{m}_q}^{\overline{m}_q-2} c^a \cdot r^r_{qi} + \sum_{q=2}^{p+1} \sum_{i=\underline{m}_q}^{\overline{m}_q-1} c^a \cdot r^d_{qi} + \sum_{q=1}^{p} \sum_{i=\underline{m}_q}^{\overline{m}_q-1} c^a \cdot r^u_{qi} + \sum_{q=1}^{p} \sum_{\substack{i=\underline{m}_q-1,\\ i\geq \underline{m}_{q-1}}}^{\underline{m}_q-1} c^a \cdot r^u_{qi} + \sum_{q=1}^{p+1} \sum_{i=\underline{m}_q+2}^{\overline{m}_q} c^a \cdot l^l_{qi} + c^a \cdot l^l_{1,\underline{m}_1+1} \\
& + \sum_{q=2}^{p+1} \sum_{i=\underline{m}_q+1}^{\overline{m}_q} c^a \cdot l^d_{qi} + \sum_{q=1}^{p} \sum_{i=\underline{m}_q+1}^{\overline{m}_q} c^a \cdot l^u_{qi} + \sum_{q=1}^{p} \sum_{\substack{i=\overline{m}_q+1,\\ i\leq \overline{m}_{q-1}}}^{\overline{m}_q+1} c^a \cdot l^u_{qi} + \sum_{q=1}^{p-1} \sum_{\substack{i\in I^q_1,\\ i\leq \overline{m}_{q+1}}} \overline{c} \cdot v^u_{qi} + \sum_{q=1}^{p} \sum_{i\in I^q_1\setminus\{\overline{m}_q\}} \overline{c} \cdot v^{u,r}_{qi} \\
& + \sum_{q=1}^{p} \sum_{i\in I^q_1\setminus\{\underline{m}_q\}} \overline{c} \cdot v^{u,l}_{qi} + \sum_{q=1}^{p} \sum_{i\in I^q\setminus I^q_1} (c^{e,d}_{qi1} \cdot e^d_{qi1} + c^{e,u}_{qi1} \cdot e^u_{qi1}) + \sum_{q=1}^{p} \sum_{i\in I^q_3\cup I^q_5} c^{e,d}_{qi2} \cdot e^d_{qi2} + \sum_{q=1}^{p} \sum_{i\in I^q_0\cup I^q_4} c^{e,d}_{qi3} \cdot e^d_{qi3} \\
& + \sum_{q=1}^{p} \sum_{i\in I^q_3\cup I^q_4} c^{e,u}_{qi2} \cdot e^u_{qi2} + \sum_{q=1}^{p} \sum_{i\in I^q_0\cup I^q_5} c^{e,u}_{qi3} \cdot e^u_{qi3} + \sum_{q=1}^{p} \sum_{i\in I^q\setminus I^q_1} (c^{t,d}_{qi} \cdot t^d_{qi} + c^{t,u}_{qi} \cdot t^u_{qi}) + \sum_{q=1}^{p} \sum_{i\in I^q_0} \sum_{s=1}^{3} (c^{w,u}_{qis} \cdot w^u_{qis} + c^{w,d}_{qis} \cdot w^d_{qis}) \\
& + \sum_{q=1}^{p} \sum_{i\in I^q_3} (c^{w,u}_{qi1} \cdot w^u_{qi1} + c^{w,d}_{qi1} \cdot w^d_{qi1}) + \sum_{q=1}^{p} \sum_{i\in I^q_4\cup I^q_5} \sum_{s=1}^{2} (c^{w,u}_{qis} \cdot w^u_{qis} + c^{w,d}_{qis} \cdot w^d_{qis}) + \sum_{q=1}^{p-1} \sum_{\substack{i\in I^q_2,\\ i\leq \overline{m}_{q+1}}} c^{e,u}_{qi1} \cdot v^u_{qi} \\
& + \sum_{q=1}^{p-1} \sum_{\substack{i\in I^q\setminus(I^q_1\cup I^q_2):\\ i\leq \overline{m}_{q+1}}} c^{t,d}_{qi} \cdot v^u_{qi} + \sum_{q=2}^{p} \sum_{i\in I^q_2} c^{e,d}_{qi1} \cdot v^d_{qi} + \sum_{q=2}^{p} \sum_{i\in I^q\setminus(I^q_1\cup I^q_2)} c^{t,u}_{qi} \cdot v^d_{qi} + \sum_{q=1}^{p} \sum_{i\in I^q_2\setminus\{\overline{m}_q\}} c^{e,u}_{qi1} \cdot v^{u,r}_{qi} \\
& + \sum_{q=1}^{p} \sum_{i\in I^q\setminus(I^q_1\cup I^q_2\cup\{\overline{m}_q\})} c^{t,d}_{qi} \cdot v^{u,r}_{qi} + \sum_{q=1}^{p} \sum_{i\in I^q_2\setminus\{\overline{m}_{q-1}\}} c^{e,d}_{qi1} \cdot v^{d,r}_{qi} + \sum_{q=1}^{p} \sum_{i\in I^q\setminus(I^q_1\cup I^q_2\cup\{\overline{m}_{q-1}\})} c^{t,u}_{qi} \cdot v^{d,r}_{qi} + \sum_{q=1}^{p} \sum_{i\in I^q_2\setminus\{\underline{m}_q\}} c^{e,u}_{qi1} \cdot v^{u,l}_{qi} \\
& + \sum_{q=1}^{p} \sum_{i\in I^q\setminus(I^q_1\cup I^q_2\cup\{\underline{m}_q\})} c^{t,d}_{qi} \cdot v^{u,l}_{qi} + \sum_{q=1}^{p} \sum_{i\in I^q_2\setminus\{\underline{m}_{q-1}\}} c^{e,d}_{qi1} \cdot v^{d,l}_{qi} + \sum_{q=1}^{p} \sum_{i\in I^q\setminus(I^q_1\cup I^q_2\cup\{\underline{m}_{q-1}\})} c^{t,u}_{qi} \cdot v^{d,l}_{qi} + \sum_{\substack{i\in I^1_2,\\ i=1}} c^{e,d}_{111} \cdot v^{d,l}_{11} \\
& + \sum_{\substack{i\in I^1\setminus(I^1_1\cup I^1_2):\\ i=1}} c^{t,u}_{11} \cdot v^{d,l}_{11} + c^0 \cdot (y^0_l + y^0_r + y^0_u + y^l_0) + \sum_{q=2}^{p} \sum_{i\in I^q_1} \overline{c} \cdot v^d_{qi} + \sum_{q=1}^{p} \sum_{\substack{i\in I^q_1:\\ i<\overline{m}_{q-1}}} \overline{c} \cdot v^{d,r}_{qi} + \sum_{q=1}^{p} \sum_{\substack{i\in I^q_1:\\ i>\underline{m}_{q-1}}} \overline{c} \cdot v^{d,l}_{qi} + \sum_{\substack{i\in I^1_1:\\ i=1}} \overline{c} \cdot v^{d,l}_{11} \qquad (1)
\end{aligned}
$$

Depot Inclusion Constraint:

$$y^0_l + y^0_r + y^0_u \geq 1 \qquad (2)$$

Item Inclusion Constraints:

$$w^u_{qi1} + w^d_{qi3} \geq 1 \qquad \forall\ q \in B, i \in I^q_0 \qquad (3)$$

$$w^u_{qi3} + w^d_{qi1} \geq 1 \qquad \forall\ q \in B, i \in I^q_0 \qquad (4)$$

$$t^u_{qi} + e^d_{qi1} + t^d_{qi} + e^u_{qi1} \geq 1 \qquad \forall\ q \in B, i \in I^q_2 \qquad (5)$$

$$w^u_{qi1} + w^d_{qi1} + e^d_{qi2} \geq 1 \qquad \forall\ q \in B, i \in I^q_3 \qquad (6)$$

$$w^u_{qi1} + w^d_{qi1} + e^u_{qi2} \geq 1 \qquad \forall\, q \in B, i \in I^q_3 \tag{7}$$

$$w^u_{qi1} + w^d_{qi2} + e^d_{qi3} \geq 1 \qquad \forall\, q \in B, i \in I^q_4 \tag{8}$$

$$w^u_{qi2} + w^d_{qi1} \geq 1 \qquad \forall\, q \in B, i \in I^q_4 \tag{9}$$

$$w^u_{qi1} + w^d_{qi2} \geq 1 \qquad \forall\, q \in B, i \in I^q_5 \tag{10}$$

$$w^u_{qi2} + e^u_{qi3} + w^d_{qi1} \geq 1 \qquad \forall\, q \in B, i \in I^q_5 \tag{11}$$

Degree Constraints:

- Constraint corresponding to the depot

$$y^0_l + y^0_r + y^0_u = y^l_0 \tag{12}$$

- Constraints corresponding to vertices $[r, q, i]$

$$r^r_{1i} + r^u_{1i} = r^r_{1,i-1} + v^{d,r}_{1i} \qquad \forall\, i \in I^1 \setminus \{\underline{m}_1, \overline{m}_1 - 1, \overline{m}_1\} \tag{13}$$

$$r^r_{1,\underline{m}_1} + r^u_{1\underline{m}_1} = y^0_r + v^{d,r}_{1,\underline{m}_1} \tag{14}$$

$$r^u_{1,\overline{m}_1-1} = r^r_{1,\overline{m}_1-1} + v^{d,r}_{1,\overline{m}_1-1} \tag{15}$$

$$r^r_{qi} + r^u_{qi} + r^d_{qi} = r^r_{q,i-1} + v^{u,r}_{q-1,i} + v^{d,r}_{qi} \qquad \forall\, q \in F \setminus \{1, p+1\}, i \in I^q \setminus \{\underline{m}_{q-1}, \overline{m}_{q-1} - 1, \overline{m}_{q-1}\} \text{ with } \underline{m}_q - 1 < i < \overline{m}_q \tag{16}$$

$$r^r_{qi} + r^d_{qi} = r^r_{q,i-1} + v^{u,r}_{q-1,i} \qquad \forall\, q \in F \setminus \{1, p+1\}, i \in I^q \setminus \{\underline{m}_{q-1}, \overline{m}_{q-1} - 1, \overline{m}_{q-1}\} \text{ with } i < \underline{m}_q - 1 \text{ or } i > \overline{m}_q \tag{17}$$

$$r^r_{qi} + r^d_{qi} = r^r_{q,i-1} + v^{u,r}_{q-1,i} + v^{d,r}_{qi} \qquad \forall\, q \in F \setminus \{1, p+1\}, i \in I^q \setminus \{\underline{m}_{q-1}, \overline{m}_{q-1} - 1, \overline{m}_{q-1}\} \text{ with } i = \overline{m}_q \tag{18}$$

$$r^r_{qi} + r^u_{qi} + r^d_{qi} = r^r_{q,i-1} + v^{u,r}_{q-1,i} \qquad \forall\, q \in F \setminus \{1, p+1\}, i \in I^q \setminus \{\underline{m}_{q-1}, \overline{m}_{q-1} - 1, \overline{m}_{q-1}\} \text{ with } i = \underline{m}_q - 1 \tag{19}$$

$$r^r_{q,\underline{m}_{q-1}} + r^u_{q,\underline{m}_{q-1}} + r^d_{q,\underline{m}_{q-1}} = v^{u,r}_{q-1,\underline{m}_{q-1}} + v^{d,r}_{q,\underline{m}_{q-1}} \qquad \forall\, q \in F \setminus \{1, p+1\} \text{ with } \underline{m}_q = \underline{m}_{q-1} \tag{20}$$

$$r^r_{q,\underline{m}_{q-1}} + r^u_{q,\underline{m}_{q-1}} + r^d_{q,\underline{m}_{q-1}} = v^{u,r}_{q-1,\underline{m}_{q-1}} \qquad \forall\, q \in F \setminus \{1, p+1\} \text{ with } \underline{m}_q = \underline{m}_{q-1} + 1 \tag{21}$$

$$r^r_{q,\underline{m}_{q-1}} + r^d_{q,\underline{m}_{q-1}} = v^{u,r}_{q-1,\underline{m}_{q-1}} \qquad \forall\, q \in F \setminus \{1, p+1\} \text{ with } \underline{m}_q > \underline{m}_{q-1} + 1 \tag{22}$$

$$r^u_{q,\overline{m}_{q-1}-1} + r^d_{q,\overline{m}_{q-1}-1} = r^r_{q,\overline{m}_{q-1}-2} + v^{u,r}_{q-1,\overline{m}_{q-1}-1} + v^{d,r}_{q,\overline{m}_{q-1}-1} \qquad \forall\, q \in F \setminus \{1, p+1\} \text{ with } \overline{m}_q = \overline{m}_{q-1} \tag{23}$$

$$r^d_{q,\overline{m}_{q-1}-1} = r^r_{q,\overline{m}_{q-1}-2} + v^{u,r}_{q-1,\overline{m}_{q-1}-1} + v^{d,r}_{q,\overline{m}_{q-1}-1} \qquad \forall\, q \in F \setminus \{1, p+1\} \text{ with } \overline{m}_q = \overline{m}_{q-1} - 1 \tag{24}$$

$$r^d_{q,\overline{m}_{q-1}-1} = r^r_{q,\overline{m}_{q-1}-2} + v^{u,r}_{q-1,\overline{m}_{q-1}-1} \qquad \forall\, q \in F \setminus \{1, p+1\} \text{ with } \overline{m}_q < \overline{m}_{q-1} - 1 \tag{25}$$

$$r^r_{p+1,i} + r^d_{p+1,i} = r^r_{p+1,i-1} + v^{u,r}_{pi} \qquad \forall\, i \in I^p \setminus \{\underline{m}_p, \overline{m}_p - 1, \overline{m}_p\} \tag{26}$$

$$r^r_{p+1,\underline{m}_p} + r^d_{p+1,\underline{m}_p} = v^{u,r}_{p,\underline{m}_p} \tag{27}$$

$$r^d_{p+1,\overline{m}_p-1} = r^r_{p+1,\overline{m}_p-2} + v^{u,r}_{p,\overline{m}_p-1} \tag{28}$$

- Constraints corresponding to vertices $[l, q, i]$

$$l^l_{1i} + l^u_{1i} = l^l_{1,i+1} + v^{d,l}_{1i} \qquad \forall\, i \in I^1 \setminus \{\underline{m}_1, \overline{m}_1\} \tag{29}$$

$$y^l_0 = y^0_l + l^l_{1,\underline{m}_1+1} + v^{d,l}_{1,\underline{m}_1} \tag{30}$$

$$l^l_{1,\overline{m}_1} + l^u_{1,\overline{m}_l} = v^{d,l}_{1,\overline{m}_1} \tag{31}$$

$$l^l_{qi} + l^u_{qi} + l^d_{qi} = l^l_{q,i+1} + v^{d,l}_{qi} + v^{u,l}_{q-1,i} \qquad \forall\, q \in F \setminus \{1, p+1\}, i \in I^q \setminus \{\underline{m}_{q-1}, \underline{m}_{q-1} + 1, \overline{m}_{q-1}\} \text{ with } \underline{m}_q < i < \overline{m}_q + 1 \tag{32}$$

$$l^l_{qi} + l^d_{qi} = l^l_{q,i+1} + v^{u,l}_{q-1,i} \qquad \forall\, q \in F\backslash\{1,p+1\}, i \in I^q\backslash\{\underline{m}_{q-1}, \underline{m}_{q-1}+1, \overline{m}_{q-1}\} \text{ with } i < \underline{m}_q \text{ or } i > \overline{m}_q + 1 \tag{33}$$

$$l^l_{qi} + l^d_{qi} = l^l_{q,i+1} + v^{d,l}_{qi} + v^{u,l}_{q-1,i} \qquad \forall\, q \in F\backslash\{1,p+1\}, i \in I^q\backslash\{\underline{m}_{q-1}, \underline{m}_{q-1}+1, \overline{m}_{q-1}\} \text{ with } i = \underline{m}_q \tag{34}$$

$$l^l_{qi} + l^u_{qi} + l^d_{qi} = l^l_{q,i+1} + v^{u,l}_{q-1,i} \qquad \forall\, q \in F\backslash\{1,p+1\}, i \in I^q\backslash\{\underline{m}_{q-1}, \underline{m}_{q-1}+1, \overline{m}_{q-1}\} \text{ with } i = \overline{m}_q + 1 \tag{35}$$

$$l^u_{q,\underline{m}_{q-1}+1} + l^d_{q,\underline{m}_{q-1}+1} = l^l_{q,\underline{m}_{q-1}+2} + v^{d,l}_{q,\underline{m}_{q-1}+1} + v^{u,l}_{q-1,\underline{m}_{q-1}+1} \qquad \forall\, q \in F\backslash\{1,p+1\} \text{ with } \underline{m}_q = \underline{m}_{q-1} \tag{36}$$

$$l^d_{q,\underline{m}_{q-1}+1} = l^l_{q,\underline{m}_{q-1}+2} + v^{d,l}_{q,\underline{m}_{q-1}+1} + v^{u,l}_{q-1,\underline{m}_{q-1}+1} \tag{37}$$

$$l^d_{q,\underline{m}_{q-1}+1} = l^l_{q,\underline{m}_{q-1}+2} + v^{u,l}_{q-1,\underline{m}_{q-1}+1} \qquad \forall\, q \in F\backslash\{1,p+1\} \text{ with } \underline{m}_q > \underline{m}_{q-1} + 1 \tag{38}$$

$$l^l_{q,\overline{m}_{q-1}} + l^u_{q,\overline{m}_{q-1}} + l^d_{q,\overline{m}_{q-1}} = v^{d,l}_{q,\overline{m}_{q-1}} + v^{u,l}_{q-1,\overline{m}_{q-1}} \qquad \forall\, q \in F\backslash\{1,p+1\} \text{ with } \overline{m}_q = \overline{m}_{q-1} \tag{39}$$

$$l^l_{q,\overline{m}_{q-1}} + l^u_{q,\overline{m}_{q-1}} + l^d_{q,\overline{m}_{q-1}} = v^{u,l}_{q-1,\overline{m}_{q-1}} \qquad \forall\, q \in F\backslash\{1,p+1\} \text{ with } \overline{m}_q = \overline{m}_{q-1} - 1 \tag{40}$$

$$l^l_{q,\overline{m}_{q-1}} + l^d_{q,\overline{m}_{q-1}} = v^{u,l}_{q-1,\overline{m}_{q-1}} \qquad \forall\, q \in F\backslash\{1,p+1\} \text{ with } \overline{m}_q < \overline{m}_{q-1} - 1 \tag{41}$$

$$l^l_{p+1,i} + l^d_{p+1,i} = l^l_{p+1,i+1} + v^{u,l}_{pi} \qquad \forall\, i \in I^p\backslash\{\underline{m}_p, \underline{m}_p+1, \overline{m}_p\} \tag{42}$$

$$l^d_{p+1,\underline{m}_p+1} = l^l_{p+1,\underline{m}_p+2} + v^{u,l}_{p,\underline{m}_p+1} \tag{43}$$

$$l^l_{p+1,\overline{m}_p} + l^d_{p+1,\overline{m}_p} = v^{u,l}_{p,\overline{m}_p} \tag{44}$$

- Constraints corresponding to vertices $[u,q,i]$

$$v^u_{qi} + v^{u,l}_{qi} + v^{u,r}_{qi} = l^u_{q,i+1} + r^u_{q,i-1} + v^u_{q-1,i} \qquad \forall\, q \in B\backslash\{1,p\}, i \in I^q_1\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } \underline{m}_{q+1} \le i \le \overline{m}_{q+1} \tag{45}$$

$$v^u_{1,\underline{m}_1} + v^{u,r}_{1,\underline{m}_1} = l^u_{1,\underline{m}_1+1} + y^0_u \qquad \text{if } \underline{m}_1 = \underline{m}_2 \text{ and } \underline{m}_1 \in I^1_1 \tag{46}$$

$$v^u_{q,\overline{m}_q} + v^{u,l}_{q,\overline{m}_q} = r^u_{q,\overline{m}_q-1} + v^u_{q-1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^q_1 \text{ and } \overline{m}_q = \overline{m}_{q+1} \text{ and } \overline{m}_q = \overline{m}_{q-1} \tag{47}$$

$$v^u_{q,\overline{m}_q} + v^{u,l}_{q,\overline{m}_q} = r^u_{q,\overline{m}_q-1} + l^u_{q,\overline{m}_q+1} + v^u_{q-1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^q_1 \text{ and } \overline{m}_q = \overline{m}_{q+1} \text{ and } \overline{m}_q \neq \overline{m}_{q-1} \tag{48}$$

$$v^{u,l}_{qi} + v^{u,r}_{qi} = l^u_{q,i+1} + r^u_{q,i-1} + v^u_{q-1,i} \qquad \forall\, q \in B\backslash\{1,p\}, i \in I^q_1\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \tag{49}$$

$$v^{u,r}_{1,\underline{m}_1} = l^u_{1,\underline{m}_1+1} + y^0_u \qquad \text{if } \underline{m}_1 \neq \underline{m}_2 \text{ and } \underline{m}_1 \in I^1_1 \tag{50}$$

$$v^{u,l}_{q,\overline{m}_q} = r^u_{q,\overline{m}_q-1} + v^u_{q-1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^q_1 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \text{ and } \overline{m}_q = \overline{m}_{q-1} \tag{51}$$

$$v^{u,l}_{q,\overline{m}_q} = r^u_{q,\overline{m}_q-1} + l^u_{q,\overline{m}_q+1} + v^u_{q-1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^q_1 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \text{ and } \overline{m}_q \neq \overline{m}_{q-1} \tag{52}$$

$$v^u_{q,\underline{m}_q} + v^{u,r}_{q,\underline{m}_q} = l^u_{q,\underline{m}_q+1} + v^u_{q-1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \in I^q_1 \text{ and } \underline{m}_q = \underline{m}_{q+1} \text{ and } \underline{m}_q = \underline{m}_{q-1} \tag{53}$$

$$v^u_{q,\underline{m}_q} + v^{u,r}_{q,\underline{m}_q} = r^u_{q,\underline{m}_q-1} + l^u_{q,\underline{m}_q+1} + v^u_{q-1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \in I^q_1 \text{ and } \underline{m}_q = \underline{m}_{q+1} \text{ and } \underline{m}_q \neq \underline{m}_{q-1} \tag{54}$$

$$v^{u,r}_{q,\underline{m}_q} = l^u_{q,\underline{m}_q+1} + v^u_{q-1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \in I^q_1 \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \text{ and } \underline{m}_q = \underline{m}_{q-1} \tag{55}$$

$$v^{u,r}_{q,\underline{m}_q} = r^u_{q,\underline{m}_q-1} + l^u_{q,\underline{m}_q+1} + v^u_{q-1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \in I^q_1 \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \text{ and } \underline{m}_q \neq \underline{m}_{q-1} \tag{56}$$

$$v^{u,l}_{pi} + v^{u,r}_{pi} = l^u_{p,i+1} + r^u_{p,i-1} + v^u_{p-1,i} \qquad \forall\, i \in I^p_1\backslash\{\underline{m}_p, \overline{m}_p\} \tag{57}$$

$$v^{u,r}_{p,\underline{m}_p} = l^u_{p,\underline{m}_p+1} + v^u_{p-1,\underline{m}_p} \qquad \text{if } \underline{m}_p \in I^p_1 \tag{58}$$

$$v^{u,l}_{p,\overline{m}_p} = r^u_{p,\overline{m}_p-1} + v^u_{p-1,\overline{m}_p} \qquad \text{if } \overline{m}_p \in I^p_1 \tag{59}$$

$$v^u_{1i} + v^{u,l}_{1i} + v^{u,r}_{1i} = l^u_{1,i+1} + r^u_{1,i-1} \qquad \forall\, i \in I^1_1\backslash\{\underline{m}_1, \overline{m}_1\} \text{ with } \underline{m}_2 \le i \le \overline{m}_2 \tag{60}$$

$$v^{u,l}_{1i} + v^{u,r}_{1i} = l^u_{1,i+1} + r^u_{1,i-1} \qquad \forall\, i \in I^1_1\backslash\{\underline{m}_1, \overline{m}_1\} \text{ with } i < \underline{m}_2 \text{ or } i > \overline{m}_2 \tag{61}$$

$$v^u_{1,\overline{m}_1} + v^{u,l}_{1,\overline{m}_1} = r^u_{1,\overline{m}_1-1} \qquad \text{if } \overline{m}_1 = \overline{m}_2 \text{ and } \overline{m}_1 \in I^1_1 \tag{62}$$

$$v^{u,l}_{1,\overline{m}_1} = r^u_{1,\overline{m}_1-1} \qquad \text{if } \overline{m}_1 \neq \overline{m}_2 \text{ and } \overline{m}_1 \in I^1_1 \tag{63}$$

$$t^u_{1i} + e^d_{1i1} = l^u_{1,i+1} + r^u_{1,i-1} \qquad \forall i \in I^1_2 \backslash \{\underline{m}_1, \overline{m}_1\} \tag{64}$$

$$t^u_{1,\underline{m}_1} + e^d_{1,\underline{m}_1,1} = l^u_{1,\underline{m}_1+1} + y^0_u \qquad \text{if } \underline{m}_1 \in I^1_2 \tag{65}$$

$$t^u_{1,\overline{m}_1} + e^d_{1,\overline{m}_1,1} = r^u_{1,\overline{m}_1-1} \qquad \text{if } \overline{m}_1 \in I^1_2 \tag{66}$$

$$t^u_{qi} + e^d_{qi1} = l^u_{q,i+1} + r^u_{q,i-1} + v^u_{q-1,i} \qquad \forall\, q \in B\backslash\{1\}, i \in I^q_2 \backslash \{\underline{m}_q, \overline{m}_q\} \tag{67}$$

$$t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} = l^u_{q,\underline{m}_q+1} + v^u_{q-1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_2 \text{ and } \underline{m}_q = \underline{m}_{q-1} \tag{68}$$

$$t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} = r^u_{q,\underline{m}_q-1} + l^u_{q,\underline{m}_q+1} + v^u_{q-1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_2 \text{ and } \underline{m}_q \neq \underline{m}_{q-1} \tag{69}$$

$$t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} = r^u_{q,\overline{m}_q-1} + v^u_{q-1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_2 \text{ and } \overline{m}_q = \overline{m}_{q-1} \tag{70}$$

$$t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} = r^u_{q,\overline{m}_q-1} + l^u_{q,\overline{m}_q+1} + v^u_{q-1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_2 \text{ and } \overline{m}_q \neq \overline{m}_{q-1} \tag{71}$$

$$t^u_{1i} + e^d_{1i1} + e^d_{1i3} = l^u_{1,i+1} + r^u_{1,i-1} \qquad \forall\, i \in (I^1_0 \cup I^1_4) \backslash \{\underline{m}_1, \overline{m}_1\} \tag{72}$$

$$t^u_{1,\underline{m}_1} + e^d_{1,\underline{m}_1,1} + e^d_{1,\underline{m}_1,3} = l^u_{1,\underline{m}_1+1} + y^0_u \qquad \text{if } \underline{m}_1 \in I^1_0 \cup I^1_4 \tag{73}$$

$$t^u_{1,\overline{m}_1} + e^d_{1,\overline{m}_1,1} + e^d_{1,\overline{m}_1,3} = r^u_{1,\overline{m}_1-1} \qquad \text{if } \overline{m}_1 \in I^1_0 \cup I^1_4 \tag{74}$$

$$t^u_{qi} + e^d_{qi1} + e^d_{qi3} = l^u_{q,i+1} + r^u_{q,i-1} + v^u_{q-1,i} \qquad \forall\, q \in B\backslash\{1\}, i \in (I^q_2 \cup I^q_4) \backslash \{\underline{m}_q, \overline{m}_q\} \tag{75}$$

$$t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} + e^d_{q,\underline{m}_q,3} = l^u_{q,\underline{m}_q+1} + v^u_{q-1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_0 \cup I^q_4 \text{ and } \underline{m}_q = \underline{m}_{q-1} \tag{76}$$

$$t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} + e^d_{q,\underline{m}_q,3} = r^u_{q,\underline{m}_q-1} + l^u_{q,\underline{m}_q+1} + v^u_{q-1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_0 \cup I^q_4 \text{ and } \underline{m}_q \neq \underline{m}_{q-1} \tag{77}$$

$$t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} + e^d_{q,\overline{m}_q,3} = r^u_{q,\overline{m}_q-1} + v^u_{q-1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_0 \cup I^q_4 \text{ and } \overline{m}_q = \overline{m}_{q-1} \tag{78}$$

$$t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} + e^d_{q,\overline{m}_q,3} = r^u_{q,\overline{m}_q-1} + l^u_{q,\overline{m}_q+1} + v^u_{q-1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_0 \cup I^q_4 \text{ and } \overline{m}_q \neq \overline{m}_{q-1} \tag{79}$$

$$t^u_{1i} + e^d_{1i1} + e^d_{1i2} = l^u_{1,i+1} + r^u_{1,i-1} \qquad \forall\, i \in (I^1_3 \cup I^1_5) \backslash \{\underline{m}_1, \overline{m}_1\} \tag{80}$$

$$t^u_{1,\underline{m}_1} + e^d_{1,\underline{m}_1,1} + e^d_{1,\underline{m}_1,2} = l^u_{1,\underline{m}_1+1} + y^0_u \qquad \text{if } \underline{m}_1 \in I^1_3 \cup I^1_5 \tag{81}$$

$$t^u_{1,\overline{m}_1} + e^d_{1,\overline{m}_1,1} + e^d_{1,\overline{m}_1,2} = r^u_{1,\overline{m}_1-1} \qquad \text{if } \overline{m}_1 \in I^1_3 \cup I^1_5 \tag{82}$$

$$t^u_{qi} + e^d_{qi1} + e^d_{qi2} = l^u_{q,i+1} + r^u_{q,i-1} + v^u_{q-1,i} \qquad \forall\, q \in B\backslash\{1\}, i \in (I^q_3 \cup I^q_5) \backslash \{\underline{m}_q, \overline{m}_q\} \tag{83}$$

$$t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} + e^d_{q,\underline{m}_q,2} = l^u_{q,\underline{m}_q+1} + v^u_{q-1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_3 \cup I^q_5 \text{ and } \underline{m}_q = \underline{m}_{q-1} \tag{84}$$

$$t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} + e^d_{q,\underline{m}_q,2} = r^u_{q,\underline{m}_q-1} + l^u_{q,\underline{m}_q+1} + v^u_{q-1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_3 \cup I^q_5 \text{ and } \underline{m}_q \neq \underline{m}_{q-1} \tag{85}$$

$$t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} + e^d_{q,\overline{m}_q,2} = r^u_{q,\overline{m}_q-1} + v^u_{q-1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_3 \cup I^q_5 \text{ and } \overline{m}_q = \overline{m}_{q-1} \tag{86}$$

$$t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} + e^d_{q,\overline{m}_q,2} = r^u_{q,\overline{m}_q-1} + l^u_{q,\overline{m}_q+1} + v^u_{q-1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_3 \cup I^q_5 \text{ and } \overline{m}_q \neq \overline{m}_{q-1} \tag{87}$$

- Constraints corresponding to vertices $[d,q,i]$

$$v^{d,l}_{1i} + v^{d,r}_{1i} = l^d_{2,i+1} + r^d_{2,i-1} + v^d_{2i} \qquad \forall\, i \in I^1_1 \backslash \{\underline{m}_1, \overline{m}_1\} \text{ with } \underline{m}_2 \leq i \leq \overline{m}_2 \tag{88}$$

$$v^{d,l}_{1i} + v^{d,r}_{1i} = l^d_{2,i+1} + r^d_{2,i-1} \qquad \forall\, i \in I^1_1 \backslash \{\underline{m}_1, \overline{m}_1\} \text{ with } i < \underline{m}_2 \text{ or } i > \overline{m}_2 \tag{89}$$

$$v^{d,l}_{1,\underline{m}_1} + v^{d,r}_{1,\underline{m}_1} = l^d_{2,\underline{m}_1+1} + v^d_{2,\underline{m}_1} \qquad \text{if } \underline{m}_1 \in I^1_1 \text{ and } \underline{m}_1 = \underline{m}_2 \tag{90}$$

$$v^{d,l}_{1,\underline{m}_1} + v^{d,r}_{1,\underline{m}_1} = l^d_{2,\underline{m}_1+1} \qquad \text{if } \underline{m}_1 \in I^1_1 \text{ and } \underline{m}_1 \neq \underline{m}_2 \tag{91}$$

$$v^{d,l}_{1,\overline{m}_1} = r^d_{2,\overline{m}_1-1} + v^d_{2,\overline{m}_1} \qquad \text{if } \overline{m}_1 \in I^1_1 \text{ and } \overline{m}_1 = \overline{m}_2 \tag{92}$$

$$v^{d,l}_{1,\overline{m}_1} = r^d_{2,\overline{m}_1-1} \qquad \text{if } \overline{m}_1 \in I^1_1 \text{ and } \overline{m}_1 \neq \overline{m}_2 \tag{93}$$

$$v^{d,l}_{qi} + v^{d,r}_{qi} + v^d_{qi} = l^d_{q+1,i+1} + r^d_{q+1,i-1} + v^d_{q+1,i} \qquad \forall\, q \in B\backslash\{1,p\}, i \in I^q_1 \backslash \{\underline{m}_q, \overline{m}_q\} \text{ with } \underline{m}_{q+1} \leq 1 \leq \overline{m}_{q+1} \tag{94}$$

$$v^{d,l}_{qi} + v^{d,r}_{qi} + v^{d}_{qi} = l^{d}_{q+1,i+1} + r^{d}_{q+1,i-1} \qquad \forall\, q \in B\backslash\{1,p\}, i \in I^{q}_{1}\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \tag{95}$$

$$v^{d,r}_{q,\underline{m}_q} + v^{d}_{q,\underline{m}_q} = l^{d}_{q+1,\underline{m}_q+1} + v^{d}_{q+1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \in I^{q}_{1} \text{ and } \underline{m}_q = \underline{m}_{q+1}, \underline{m}_q = \underline{m}_{q-1} \tag{96}$$

$$v^{d,r}_{q,\underline{m}_q} + v^{d}_{q,\underline{m}_q} = l^{d}_{q+1,\underline{m}_q+1} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \in I^{q}_{1} \text{ and } \underline{m}_q \neq \underline{m}_{q+1}, \underline{m}_q = \underline{m}_{q-1} \tag{97}$$

$$v^{d,l}_{q,\overline{m}_q} + v^{d}_{q,\overline{m}_q} = r^{d}_{q+1,\overline{m}_q-1} + v^{d}_{q+1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^{q}_{1} \text{ and } \overline{m}_q = \overline{m}_{q+1}, \overline{m}_q = \overline{m}_{q-1} \tag{98}$$

$$v^{d,l}_{q,\overline{m}_q} + v^{d}_{q,\overline{m}_q} = r^{d}_{q+1,\overline{m}_q-1} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^{q}_{1} \text{ and } \overline{m}_q \neq \overline{m}_{q+1}, \overline{m}_q = \overline{m}_{q-1} \tag{99}$$

$$v^{d,l}_{pi} + v^{d,r}_{pi} + v^{d}_{pi} = l^{d}_{p+1,i+1} + r^{d}_{p+1,i-1} \qquad \forall\, i \in I^{p}_{1}\backslash\{\underline{m}_p, \overline{m}_p\} \tag{100}$$

$$v^{d,l}_{p,\underline{m}_p} + v^{d,r}_{p,\underline{m}_p} + v^{d}_{p,\underline{m}_p} = l^{d}_{p+1,\underline{m}_p+1} \qquad \text{if } \underline{m}_p \in I^{p}_{1} \text{ and } \underline{m}_p \neq \underline{m}_{p-1} \tag{101}$$

$$v^{d,r}_{p,\underline{m}_p} + v^{d}_{p,\underline{m}_p} = l^{d}_{p+1,\underline{m}_p+1} \qquad \text{if } \underline{m}_p \in I^{p}_{1} \text{ and } \underline{m}_p = \underline{m}_{p-1} \tag{102}$$

$$v^{d,l}_{p,\overline{m}_p} + v^{d,r}_{p,\overline{m}_p} + v^{d}_{p,\overline{m}_p} = r^{d}_{p+1,\overline{m}_p-1} \qquad \text{if } \overline{m}_p \in I^{p}_{1} \text{ and } \overline{m}_p \neq \overline{m}_{p-1} \tag{103}$$

$$v^{d,l}_{p,\overline{m}_p} + v^{d}_{p,\overline{m}_p} = r^{d}_{p+1,\overline{m}_p-1} \qquad \text{if } \overline{m}_p \in I^{p}_{1} \text{ and } \overline{m}_p = \overline{m}_{p-1} \tag{104}$$

$$v^{d,l}_{q,\underline{m}_q} + v^{d,r}_{q,\underline{m}_q} + v^{d}_{q,\underline{m}_q} = l^{d}_{q+1,\underline{m}_q+1} + v^{d}_{q+1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \in I^{q}_{1} \text{ and } \underline{m}_q = \underline{m}_{q+1}, \underline{m}_q \neq \underline{m}_{q-1} \tag{105}$$

$$v^{d,l}_{q,\underline{m}_q} + v^{d,r}_{q,\underline{m}_q} + v^{d}_{q,\underline{m}_q} = l^{d}_{q+1,\underline{m}_q+1} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \in I^{q}_{1} \text{ and } \underline{m}_q \neq \underline{m}_{q+1}, \underline{m}_q \neq \underline{m}_{q-1} \tag{106}$$

$$v^{d,l}_{q,\overline{m}_q} + v^{d,r}_{q,\overline{m}_q} + v^{d}_{q,\overline{m}_q} = r^{d}_{q+1,\overline{m}_q-1} + v^{d}_{q+1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^{q}_{1} \text{ and } \overline{m}_q = \overline{m}_{q+1}, \overline{m}_q \neq \overline{m}_{q-1} \tag{107}$$

$$v^{d,l}_{q,\overline{m}_q} + v^{d,r}_{q,\overline{m}_q} + v^{d}_{q,\overline{m}_q} = r^{d}_{q+1,\overline{m}_q-1} \qquad \forall\, q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^{q}_{1} \text{ and } \overline{m}_q \neq \overline{m}_{q+1}, \overline{m}_q \neq \overline{m}_{q-1} \tag{108}$$

$$e^{u}_{qi1} + t^{d}_{qi} = l^{d}_{q+1,i+1} + r^{d}_{q+1,i-1} + v^{d}_{q+1,i} \qquad \forall\, q \in B\backslash\{p\} \text{ with } i \in I^{q}_{2}\backslash\{\underline{m}_q, \overline{m}_q\}, \underline{m}_{q+1} \leq i \leq \overline{m}_{q+1} \tag{109}$$

$$e^{u}_{qi1} + t^{d}_{qi} = l^{d}_{q+1,i+1} + r^{d}_{q+1,i-1} \qquad \forall\, q \in B\backslash\{p\} \text{ with } i \in I^{q}_{2}\backslash\{\underline{m}_q, \overline{m}_q\}, i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \tag{110}$$

$$e^{u}_{q,\underline{m}_q,1} + t^{d}_{q,\underline{m}_q} = l^{d}_{q+1,\underline{m}_q+1} + v^{d}_{q+1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^{q}_{2} \text{ and } \underline{m}_q = \underline{m}_{q+1} \tag{111}$$

$$e^{u}_{q,\underline{m}_q,1} + t^{d}_{q,\underline{m}_q} = l^{d}_{q+1,\underline{m}_q+1} \qquad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^{q}_{2} \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \tag{112}$$

$$e^{u}_{q,\overline{m}_q,1} + t^{d}_{q,\overline{m}_q} = r^{d}_{q+1,\overline{m}_q-1} + v^{d}_{q+1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^{q}_{2} \text{ and } \overline{m}_q = \overline{m}_{q+1} \tag{113}$$

$$e^{u}_{q,\overline{m}_q,1} + t^{d}_{q,\overline{m}_q} = r^{d}_{q+1,\overline{m}_q-1} \qquad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^{q}_{2} \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \tag{114}$$

$$e^{u}_{pi1} + t^{d}_{pi} = l^{d}_{p+1,i+1} + r^{d}_{p+1,i-1} \qquad \forall\, i \in I^{p}_{2}\backslash\{\underline{m}_p, \overline{m}_p\} \tag{115}$$

$$e^{u}_{p,\underline{m}_p,1} + t^{d}_{p,\underline{m}_p} = l^{d}_{p+1,\underline{m}_p+1} \qquad \text{if } \underline{m}_p \in I^{p}_{2} \tag{116}$$

$$e^{u}_{p,\overline{m}_p,1} + t^{d}_{p,\overline{m}_p} = r^{d}_{p+1,\overline{m}_p-1} \qquad \text{if } \overline{m}_p \in I^{p}_{2} \tag{117}$$

$$e^{u}_{qi1} + e^{u}_{qi2} + t^{d}_{qi} = l^{d}_{q+1,i+1} + r^{d}_{q+1,i-1} + v^{d}_{q+1,i} \qquad \forall\, q \in B\backslash\{p\}, i \in (I^{q}_{3} \cup I^{q}_{4})\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } \underline{m}_{q+1} \leq i \leq \overline{m}_{q+1} \tag{118}$$

$$e^{u}_{qi1} + e^{u}_{qi2} + t^{d}_{qi} = l^{d}_{q+1,i+1} + r^{d}_{q+1,i-1} \qquad \forall\, q \in B\backslash\{p\}, i \in (I^{q}_{3} \cup I^{q}_{4})\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \tag{119}$$

$$e^{u}_{q,\underline{m}_q,1} + e^{u}_{q,\underline{m}_q,2} + t^{d}_{q,\underline{m}_q} = l^{d}_{q+1,\underline{m}_q+1} + v^{d}_{q+1,\underline{m}_q} \qquad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^{q}_{3} \cup I^{q}_{4} \text{ and } \underline{m}_q = \underline{m}_{q+1} \tag{120}$$

$$e^{u}_{q,\underline{m}_q,1} + e^{u}_{q,\underline{m}_q,2} + t^{d}_{q,\underline{m}_q} = l^{d}_{q+1,\underline{m}_q+1} \qquad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^{q}_{3} \cup I^{q}_{4} \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \tag{121}$$

$$e^{u}_{q,\overline{m}_q,1} + e^{u}_{q,\overline{m}_q,2} + t^{d}_{q,\overline{m}_q} = r^{d}_{q+1,\overline{m}_q-1} + v^{d}_{q+1,\overline{m}_q} \qquad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^{q}_{3} \cup I^{q}_{4} \text{ and } \overline{m}_q = \overline{m}_{q+1} \tag{122}$$

$$e^{u}_{q,\overline{m}_q,1} + e^{u}_{q,\overline{m}_q,2} + t^{d}_{q,\overline{m}_q} = r^{d}_{q+1,\overline{m}_q-1} \qquad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^{q}_{3} \cup I^{q}_{4} \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \tag{123}$$

$$e^{u}_{pi1} + e^{u}_{pi2} + t^{d}_{pi} = l^{d}_{p+1,i+1} + r^{d}_{p+1,i-1} \qquad \forall\, i \in (I^{p}_{3} \cup I^{p}_{4})\backslash\{\underline{m}_p, \overline{m}_p\} \tag{124}$$

$$e^{u}_{p,\underline{m}_p,1} + e^{u}_{p,\underline{m}_p,2} + t^{d}_{p,\underline{m}_p} = l^{d}_{p+1,\underline{m}_p+1} \qquad \text{if } \underline{m}_p \in I^{p}_{3} \cup I^{p}_{4} \tag{125}$$

$$e^{u}_{p,\overline{m}_p,1} + e^{u}_{p,\overline{m}_p,2} + t^{d}_{p,\overline{m}_p} = r^{d}_{p+1,\overline{m}_p-1} \qquad \text{if } \overline{m}_p \in I^{p}_{3} \cup I^{p}_{4} \tag{126}$$

$$e^{u}_{qi1} + e^{u}_{qi3} + t^{d}_{qi} = l^{d}_{q+1,i+1} + r^{d}_{q+1,i-1} + v^{d}_{q+1,i} \qquad \forall\, q \in B\backslash\{p\}, i \in (I^{q}_{0} \cup I^{q}_{5})\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } \underline{m}_{q+1} \leq i \leq \overline{m}_{q+1} \tag{127}$$

$$e^u_{qi1} + e^u_{qi3} + t^d_{qi} = l^d_{q+1,i+1} + r^d_{q+1,i-1} \quad \forall\, q \in B\backslash\{p\}, i \in (I^q_0 \cup I^q_5)\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \tag{128}$$

$$e^u_{q,\underline{m}_q,1} + e^u_{q,\underline{m}_q,3} + t^d_{q,\underline{m}_q} = l^d_{q+1,\underline{m}_q+1} + v^d_{q+1,\underline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_0 \cup I^q_5 \text{ and } \underline{m}_q = \underline{m}_{q+1} \tag{129}$$

$$e^u_{q,\underline{m}_q,1} + e^u_{q,\underline{m}_q,3} + t^d_{q,\underline{m}_q} = l^d_{q+1,\underline{m}_q+1} \quad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_0 \cup I^q_5 \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \tag{130}$$

$$e^u_{q,\overline{m}_q,1} + e^u_{q,\overline{m}_q,3} + t^d_{q,\overline{m}_q} = r^d_{q+1,\overline{m}_q-1} + v^d_{q+1,\overline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_0 \cup I^q_5 \text{ and } \overline{m}_q = \overline{m}_{q+1} \tag{131}$$

$$e^u_{q,\overline{m}_q,1} + e^u_{q,\overline{m}_q,3} + t^d_{q,\overline{m}_q} = r^d_{q+1,\overline{m}_q-1} \quad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_0 \cup I^q_5 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \tag{132}$$

$$e^u_{pi1} + e^u_{pi3} + t^d_{pi} = l^d_{p+1,i+1} + r^d_{p+1,i-1} \quad \forall\, i \in (I^p_0 \cup I^p_5)\backslash\{\underline{m}_p, \overline{m}_p\} \tag{133}$$

$$e^u_{p,\underline{m}_p,1} + e^u_{p,\underline{m}_p,3} + t^d_{p,\underline{m}_p} = l^d_{p+1,\underline{m}_p+1} \quad \text{if } \underline{m}_p \in I^p_0 \cup I^p_5 \tag{134}$$

$$e^u_{p,\overline{m}_p,1} + e^u_{p,\overline{m}_p,3} + t^d_{p,\overline{m}_p} = r^d_{p+1,\overline{m}_p-1} \quad \text{if } \overline{m}_p \in I^p_0 \cup I^p_5 \tag{135}$$

- Constraints corresponding to vertices $[u,q,i,1]$

$$v^u_{qi} + v^{u,l}_{qi} + v^{u,r}_{qi} = e^u_{qi1} + t^u_{qi} \quad \forall\, q \in B\backslash\{p\}, i \in I^q_2\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } \underline{m}_{q+1} \leq i \leq \overline{m}_{q+1} \tag{136}$$

$$v^{u,l}_{qi} + v^{u,r}_{qi} = e^u_{qi1} + t^u_{qi} \quad \forall\, q \in B\backslash\{p\}, i \in I^q_2\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \tag{137}$$

$$v^u_{q,\underline{m}_q} + v^{u,r}_{q,\underline{m}_q} = e^u_{q,\underline{m}_q,1} + t^u_{q,\underline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_2 \text{ and } \underline{m}_q = \underline{m}_{q+1} \tag{138}$$

$$v^{u,r}_{q,\underline{m}_q} = e^u_{q,\underline{m}_q,1} + t^u_{q,\underline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_2 \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \tag{139}$$

$$v^u_{q,\overline{m}_q} + v^{u,l}_{q,\overline{m}_q} = e^u_{q,\overline{m}_q,1} + t^u_{q,\overline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_2 \text{ and } \overline{m}_q = \overline{m}_{q+1} \tag{140}$$

$$v^{u,l}_{q,\overline{m}_q} = e^u_{q,\overline{m}_q,1} + t^u_{q,\overline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_2 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \tag{141}$$

$$v^{u,l}_{pi} + v^{u,r}_{pi} = e^u_{pi1} + t^u_{pi} \quad \forall\, i \in I^p_2\backslash\{\underline{m}_p, \overline{m}_p\} \tag{142}$$

$$v^{u,r}_{p,\underline{m}_p} = e^u_{p,\underline{m}_p,1} + t^u_{p,\underline{m}_p} \quad \text{if } \underline{m}_p \in I^p_2 \tag{143}$$

$$v^{u,l}_{p,\overline{m}_p} = e^u_{p,\overline{m}_p,1} + t^u_{p,\overline{m}_p} \quad \text{if } \overline{m}_p \in I^p_2 \tag{144}$$

$$w^u_{qi1} = e^u_{qi1} + t^u_{qi} \quad \forall\, q \in B, i \in I^q\backslash(I^q_1 \cup I^q_2) \tag{145}$$

- Constraints corresponding to vertices $[u,q,i,2]$

$$v^u_{qi} + v^{u,l}_{qi} + v^{u,r}_{qi} = e^u_{qi2} + w^u_{qi1} \quad \forall\, q \in B\backslash\{p\}, i \in I^q_3 \backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } \underline{m}_{q+1} \leq i \leq \overline{m}_{q+1} \tag{146}$$

$$v^{u,l}_{qi} + v^{u,r}_{qi} = e^u_{qi2} + w^u_{qi1} \quad \forall\, q \in B\backslash\{p\}, i \in I^q_3\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \tag{147}$$

$$v^u_{q,\underline{m}_q} + v^{u,r}_{q,\underline{m}_q} = e^u_{q,\underline{m}_q,2} + w^u_{q,\underline{m}_q,1} \quad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_3 \text{ and } \underline{m}_q = \underline{m}_{q+1} \tag{148}$$

$$v^{u,r}_{q,\underline{m}_q} = e^u_{q,\underline{m}_q,2} + w^u_{q,\underline{m}_q,1} \quad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_3 \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \tag{149}$$

$$v^u_{q,\overline{m}_q} + v^{u,l}_{q,\overline{m}_q} = e^u_{q,\overline{m}_q,2} + w^u_{q,\overline{m}_q,1} \quad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_3 \text{ and } \overline{m}_q = \overline{m}_{q+1} \tag{150}$$

$$v^{u,l}_{q,\overline{m}_q} = e^u_{q,\overline{m}_q,2} + w^u_{q,\overline{m}_q,1} \quad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_3 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \tag{151}$$

$$v^{u,l}_{pi} + v^{u,r}_{pi} = e^u_{pi2} + w^u_{pi1} \quad \forall\, i \in I^p_3\backslash\{\underline{m}_p, \overline{m}_p\} \tag{152}$$

$$v^{u,r}_{p,\underline{m}_p} = e^u_{p,\underline{m}_p,2} + w^u_{p,\underline{m}_p,1} \quad \text{if } \underline{m}_p \in I^p_3 \tag{153}$$

$$v^{u,l}_{p,\overline{m}_p} = e^u_{p,\overline{m}_p,2} + w^u_{p,\overline{m}_p,1} \quad \text{if } \overline{m}_p \in I^p_3 \tag{154}$$

$$w^u_{qi2} = e^u_{qi2} + w^u_{qi1} \quad \forall\, q \in B, i \in I^q_4 \tag{155}$$

$$w^u_{qi2} = w^u_{qi1} \quad \forall\, q \in B, i \in I^q_0 \cup I^q_5 \tag{156}$$

- Constraints corresponding to vertices $[u,q,i,3]$

$$v^{u}_{qi}+v^{u,l}_{qi}+v^{u,r}_{qi}=w^{u}_{qi2} \qquad \forall\, q\in B\backslash\{p\}, i\in I^{q}_{4}\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } \underline{m}_{q+1}\le i\le \overline{m}_{q+1} \quad (157)$$

$$v^{u,l}_{qi}+v^{u,r}_{qi}=w^{u}_{qi2} \qquad \forall\, q\in B\backslash\{p\}, i\in I^{q}_{4}\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } i<\underline{m}_{q+1} \text{ or } i>\overline{m}_{q+1} \quad (158)$$

$$v^{u}_{q,\underline{m}_q}+v^{u,r}_{q,\underline{m}_q}=w^{u}_{q,\underline{m}_q,2} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^{q}_{4} \text{ and } \underline{m}_q=\underline{m}_{q+1} \quad (159)$$

$$v^{u,r}_{q,\underline{m}_q}=w^{u}_{q,\underline{m}_q,2} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^{q}_{4} \text{ and } \underline{m}_q\neq\underline{m}_{q+1} \quad (160)$$

$$v^{u}_{q,\overline{m}_q}+v^{u,l}_{q,\overline{m}_q}=w^{u}_{q,\overline{m}_q,2} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^{q}_{4} \text{ and } \overline{m}_q=\overline{m}_{q+1} \quad (161)$$

$$v^{u,l}_{q,\overline{m}_q}=w^{u}_{q,\overline{m}_q,2} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^{q}_{4} \text{ and } \overline{m}_q\neq\overline{m}_{q+1} \quad (162)$$

$$v^{u,l}_{pi}+v^{u,r}_{pi}=w^{u}_{pi2} \qquad \forall\, i\in I^{p}_{4}\backslash\{\underline{m}_p,\overline{m}_p\} \quad (163)$$

$$v^{u,r}_{p,\underline{m}_p}=w^{u}_{p,\underline{m}_p,2} \qquad \text{if } \underline{m}_p\in I^{p}_{4} \quad (164)$$

$$v^{u,l}_{p,\overline{m}_p}=w^{u}_{p,\overline{m}_p,2} \qquad \text{if } \overline{m}_p\in I^{p}_{4} \quad (165)$$

$$v^{u}_{qi}+v^{u,l}_{qi}+v^{u,r}_{qi}=e^{u}_{qi3}+w^{u}_{qi2} \qquad \forall\, q\in B\backslash\{p\}, i\in I^{q}_{5}\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } \underline{m}_{q+1}\le i\le \overline{m}_{q+1} \quad (166)$$

$$v^{u,l}_{qi}+v^{u,r}_{qi}=e^{u}_{qi3}+w^{u}_{qi2} \qquad \forall\, q\in B\backslash\{p\}, i\in I^{q}_{5}\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } i<\underline{m}_{q+1} \text{ or } i>\overline{m}_{q+1} \quad (167)$$

$$v^{u}_{q,\underline{m}_q}+v^{u,r}_{q,\underline{m}_q}=e^{u}_{q,\underline{m}_q,3}+w^{u}_{q,\underline{m}_q,2} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^{q}_{5} \text{ and } \underline{m}_q=\underline{m}_{q+1} \quad (168)$$

$$v^{u,r}_{q,\underline{m}_q}=e^{u}_{q,\underline{m}_q,3}+w^{u}_{q,\underline{m}_q,2} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^{q}_{5} \text{ and } \underline{m}_q\neq\underline{m}_{q+1} \quad (169)$$

$$v^{u}_{q,\overline{m}_q}+v^{u,l}_{q,\overline{m}_q}=e^{u}_{q,\overline{m}_q,3}+w^{u}_{q,\overline{m}_q,2} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^{q}_{5} \text{ and } \overline{m}_q=\overline{m}_{q+1} \quad (170)$$

$$v^{u,l}_{q,\overline{m}_q}=e^{u}_{q,\overline{m}_q,3}+w^{u}_{q,\overline{m}_q,2} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^{q}_{5} \text{ and } \overline{m}_q\neq\overline{m}_{q+1} \quad (171)$$

$$v^{u,l}_{pi}+v^{u,r}_{pi}=e^{u}_{pi3}+w^{u}_{pi2} \qquad \forall\, i\in I^{p}_{5}\backslash\{\underline{m}_p,\overline{m}_p\} \quad (172)$$

$$v^{u,r}_{p,\underline{m}_p}=e^{u}_{p,\underline{m}_p,3}+w^{u}_{p,\underline{m}_p,2} \qquad \text{if } \underline{m}_p\in I^{p}_{5} \quad (173)$$

$$v^{u,l}_{p,\overline{m}_p}=e^{u}_{p,\overline{m}_p,3}+w^{u}_{p,\overline{m}_p,2} \qquad \text{if } \overline{m}_p\in I^{p}_{5} \quad (174)$$

$$w^{u}_{qi3}=e^{u}_{qi3}+w^{u}_{qi2} \qquad \forall\, q\in B, i\in I^{q}_{0} \quad (175)$$

- Constraints corresponding to vertices $[u,q,i,4]$

$$v^{u}_{qi}+v^{u,l}_{qi}+v^{u,r}_{qi}=w^{u}_{qi3} \qquad \forall\, q\in B\backslash\{p\}, i\in I^{q}_{0}\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } \underline{m}_{q+1}\le i\le \overline{m}_{q+1} \quad (176)$$

$$v^{u,l}_{qi}+v^{u,r}_{qi}=w^{u}_{qi3} \qquad \forall\, q\in B\backslash\{p\}, i\in I^{q}_{0}\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } i<\underline{m}_{q+1} \text{ or } i>\overline{m}_{q+1} \quad (177)$$

$$v^{u}_{q,\underline{m}_q}+v^{u,r}_{q,\underline{m}_q}=w^{u}_{q,\underline{m}_q,3} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^{q}_{0} \text{ and } \underline{m}_q=\underline{m}_{q+1} \quad (178)$$

$$v^{u,r}_{q,\underline{m}_q}=w^{u}_{q,\underline{m}_q,3} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^{q}_{0} \text{ and } \underline{m}_q\neq\underline{m}_{q+1} \quad (179)$$

$$v^{u}_{q,\overline{m}_q}+v^{u,l}_{q,\overline{m}_q}=w^{u}_{q,\overline{m}_q,3} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^{q}_{0} \text{ and } \overline{m}_q=\overline{m}_{q+1} \quad (180)$$

$$v^{u,l}_{q,\overline{m}_q}=w^{u}_{q,\overline{m}_q,3} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^{q}_{0} \text{ and } \overline{m}_q\neq\overline{m}_{q+1} \quad (181)$$

$$v^{u,l}_{pi}+v^{u,r}_{pi}=w^{u}_{pi3} \qquad \forall\, i\in I^{p}_{0}\backslash\{\underline{m}_p,\overline{m}_p\} \quad (182)$$

$$v^{u,r}_{p,\underline{m}_p}=w^{u}_{p,\underline{m}_p,3} \qquad \text{if } \underline{m}_p\in I^{p}_{0} \quad (183)$$

$$v^{u,l}_{p,\overline{m}_p}=w^{u}_{p,\overline{m}_p,3} \qquad \text{if } \overline{m}_p\in I^{p}_{0} \quad (184)$$

- Constraints corresponding to vertices $[d,q,i,1]$

$$v_{1i}^{d,l} + v_{1i}^{d,r} = e_{1i1}^{d} + t_{1i}^{d} \qquad \forall\ i \in I_2^1 \backslash \{\overline{m}_1\} \quad (185)$$
$$v_{1,\overline{m}_1}^{d,l} = e_{1,\overline{m}_1,1}^{d} + t_{1,\overline{m}_1}^{d} \qquad \text{if } \overline{m}_1 \in I_2^1 \quad (186)$$
$$v_{qi}^{d} + v_{qi}^{d,l} + v_{qi}^{d,r} = e_{qi1}^{d} + t_{qi}^{d} \qquad \forall\ q \in B\backslash\{1\}, i \in I_2^q \backslash \{\underline{m}_q, \overline{m}_q\} \quad (187)$$
$$v_{q,\underline{m}_q}^{d} + v_{q,\underline{m}_q}^{d,l} + v_{q,\underline{m}_q}^{d,r} = e_{q,\underline{m}_q,1}^{d} + t_{q,\underline{m}_q}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I_2^q \text{ and } \underline{m}_{q-1} < \underline{m}_q \quad (188)$$
$$v_{q,\underline{m}_q}^{d} + v_{q,\underline{m}_q}^{d,r} = e_{q,\underline{m}_q,1}^{d} + t_{q,\underline{m}_q}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I_2^q \text{ and } \underline{m}_{q-1} = \underline{m}_q \quad (189)$$
$$v_{q,\overline{m}_q}^{d} + v_{q,\overline{m}_q}^{d,l} + v_{q,\overline{m}_q}^{d,r} = e_{q,\overline{m}_q,1}^{d} + t_{q,\overline{m}_q}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I_2^q \text{ and } \overline{m}_{q-1} > \overline{m}_q \quad (190)$$
$$v_{q,\overline{m}_q}^{d} + v_{q,\overline{m}_q}^{d,l} = e_{q,\overline{m}_q,1}^{d} + t_{q,\overline{m}_q}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I_2^q \text{ and } \overline{m}_{q-1} = \overline{m}_q \quad (191)$$
$$w_{qi1}^{d} = e_{qi1}^{d} + t_{qi}^{d} \qquad \forall\ q \in B, i \in I^q \backslash (I_1^q \cup I_2^q) \quad (192)$$

- Constraints corresponding to vertices $[d,q,i,2]$

$$v_{1i}^{d,l} + v_{1i}^{d,r} = e_{1i2}^{d} + w_{1i1}^{d} \qquad \forall\ i \in I_3^1 \backslash \{\overline{m}_1\} \quad (193)$$
$$v_{1,\overline{m}_1}^{d,l} = e_{1,\overline{m}_1,2}^{d} + w_{1,\overline{m}_1,1}^{d} \qquad \text{if } \overline{m}_1 \in I_3^1 \quad (194)$$
$$v_{qi}^{d} + v_{qi}^{d,l} + v_{qi}^{d,r} = e_{qi2}^{d} + w_{qi1}^{d} \qquad \forall\ q \in B\backslash\{1\}, i \in I_3^q \backslash \{\underline{m}_q, \overline{m}_q\} \quad (195)$$
$$v_{q,\underline{m}_q}^{d} + v_{q,\underline{m}_q}^{d,l} + v_{q,\underline{m}_q}^{d,r} = e_{q,\underline{m}_q,2}^{d} + w_{q,\underline{m}_q,1}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I_3^q \text{ and } \underline{m}_{q-1} < \underline{m}_q \quad (196)$$
$$v_{q,\underline{m}_q}^{d} + v_{q,\underline{m}_q}^{d,r} = e_{q,\underline{m}_q,2}^{d} + w_{q,\underline{m}_q,1}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I_3^q \text{ and } \underline{m}_{q-1} = \underline{m}_q \quad (197)$$
$$v_{q,\overline{m}_q}^{d} + v_{q,\overline{m}_q}^{d,l} + v_{q,\overline{m}_q}^{d,r} = e_{q,\overline{m}_q,2}^{d} + w_{q,\overline{m}_q,1}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I_3^q \text{ and } \overline{m}_{q-1} > \overline{m}_q \quad (198)$$
$$v_{q,\overline{m}_q}^{d} + v_{q,\overline{m}_q}^{d,l} = e_{q,\overline{m}_q,2}^{d} + w_{q,\overline{m}_q,1}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I_3^q \text{ and } \overline{m}_{q-1} = \overline{m}_q \quad (199)$$
$$w_{qi2}^{d} = w_{qi1}^{d} \qquad \forall\ q \in B, i \in I_0^q \cup I_4^q \quad (200)$$
$$w_{qi2}^{d} = e_{qi2}^{d} + w_{qi1}^{d} \qquad \forall\ q \in B, i \in I_5^q \quad (201)$$

- Constraints corresponding to vertices $[d,q,i,3]$

$$v_{1i}^{d,l} + v_{1i}^{d,r} = e_{1i3}^{d} + w_{1i2}^{d} \qquad \forall\ i \in I_4^1 \backslash \{\overline{m}_1\} \quad (202)$$
$$v_{1,\overline{m}_1}^{d,l} = e_{1,\overline{m}_1,3}^{d} + w_{1,\overline{m}_1,2}^{d} \qquad \text{if } \overline{m}_1 \in I_4^1 \quad (203)$$
$$v_{qi}^{d} + v_{qi}^{d,l} + v_{qi}^{d,r} = e_{qi3}^{d} + w_{qi2}^{d} \qquad \forall\ q \in B\backslash\{1\}, i \in I_4^q \backslash \{\underline{m}_q, \overline{m}_q\} \quad (204)$$
$$v_{q,\underline{m}_q}^{d} + v_{q,\underline{m}_q}^{d,l} + v_{q,\underline{m}_q}^{d,r} = e_{q,\underline{m}_q,3}^{d} + w_{q,\underline{m}_q,2}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I_4^q \text{ and } \underline{m}_{q-1} < \underline{m}_q \quad (205)$$
$$v_{q,\underline{m}_q}^{d} + v_{q,\underline{m}_q}^{d,r} = e_{q,\underline{m}_q,3}^{d} + w_{q,\underline{m}_q,2}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I_4^q \text{ and } \underline{m}_{q-1} = \underline{m}_q \quad (206)$$
$$v_{q,\overline{m}_q}^{d} + v_{q,\overline{m}_q}^{d,l} + v_{q,\overline{m}_q}^{d,r} = e_{q,\overline{m}_q,3}^{d} + w_{q,\overline{m}_q,2}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I_4^q \text{ and } \overline{m}_{q-1} > \overline{m}_q \quad (207)$$
$$v_{q,\overline{m}_q}^{d} + v_{q,\overline{m}_q}^{d,l} = e_{q,\overline{m}_q,3}^{d} + w_{q,\overline{m}_q,2}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I_4^q \text{ and } \overline{m}_{q-1} = \overline{m}_q \quad (208)$$
$$v_{1i}^{d,l} + v_{1i}^{d,r} = w_{1i2}^{d} \qquad \forall\ i \in I_5^1 \backslash \{\overline{m}_1\} \quad (209)$$
$$v_{1,\overline{m}_1}^{d,l} = w_{1,\overline{m}_1,2}^{d} \qquad \text{if } \overline{m}_1 \in I_5^1 \quad (210)$$
$$v_{qi}^{d} + v_{qi}^{d,l} + v_{qi}^{d,r} = w_{qi2}^{d} \qquad \forall\ q \in B\backslash\{1\}, i \in I_5^q \backslash \{\underline{m}_q, \overline{m}_q\} \quad (211)$$
$$v_{q,\underline{m}_q}^{d} + v_{q,\underline{m}_q}^{d,l} + v_{q,\underline{m}_q}^{d,r} = w_{q,\underline{m}_q,2}^{d} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I_5^q \text{ and } \underline{m}_{q-1} < \underline{m}_q \quad (212)$$

$$v^d_{q,\underline{m}_q} + v^{d,r}_{q,\underline{m}_q} = w^d_{q,\underline{m}_q,2} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_5 \text{ and } \underline{m}_{q-1} = \underline{m}_q \quad (213)$$

$$v^d_{q,\overline{m}_q} + v^{d,l}_{q,\overline{m}_q} + v^{d,r}_{q,\overline{m}_q} = w^d_{q,\overline{m}_q,2} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_5 \text{ and } \overline{m}_{q-1} > \overline{m}_q \quad (214)$$

$$v^d_{q,\overline{m}_q} + v^{d,l}_{q,\overline{m}_q} = w^d_{q,\overline{m}_q,2} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_5 \text{ and } \overline{m}_{q-1} = \overline{m}_q \quad (215)$$

$$w^d_{qi3} = e^d_{qi3} + w^d_{qi2} \qquad \forall\, q \in B, i \in I^q_0 \quad (216)$$

- Constraints corresponding to vertices $[d,q,i,4]$

$$v^{d,l}_{1i} + v^{d,r}_{1i} = w^d_{1i3} \qquad \forall\, i \in I^1_0\backslash\{\overline{m}_1\} \quad (217)$$

$$v^{d,l}_{1,\overline{m}_1} = w^d_{1,\overline{m}_1,3} \qquad \text{if } \overline{m}_1 \in I^1_0 \quad (218)$$

$$v^d_{qi} + v^{d,l}_{qi} + v^{d,r}_{qi} = w^d_{qi3} \qquad \forall\, q \in B\backslash\{1\}, i \in I^q_0\backslash\{\underline{m}_q, \overline{m}_q\} \quad (219)$$

$$v^d_{q,\underline{m}_q} + v^{d,l}_{q,\underline{m}_q} + v^{d,r}_{q,\underline{m}_q} = w^d_{q,\underline{m}_q,3} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_0 \text{ and } \underline{m}_{q-1} < \underline{m}_q \quad (220)$$

$$v^d_{q,\underline{m}_q} + v^{d,r}_{q,\underline{m}_q} = w^d_{q,\underline{m}_q,3} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_0 \text{ and } \underline{m}_{q-1} = \underline{m}_q \quad (221)$$

$$v^d_{q,\overline{m}_q} + v^{d,l}_{q,\overline{m}_q} + v^{d,r}_{q,\overline{m}_q} = w^d_{q,\overline{m}_q,3} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_0 \text{ and } \overline{m}_{q-1} > \overline{m}_q \quad (222)$$

$$v^d_{q,\overline{m}_q} + v^{d,l}_{q,\overline{m}_q} = w^d_{q,\overline{m}_q,3} \qquad \forall\, q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_0 \text{ and } \overline{m}_{q-1} = \overline{m}_q \quad (223)$$

Subtour Elimination Constraints:

- Constraints corresponding to vertices $[r,q,i]$

$$\tilde{r}^r_{1,i-1} + \tilde{v}^{d,r}_{1i} - (\tilde{r}^r_{1i} + \tilde{r}^u_{1i}) = r^r_{1i} + r^u_{1i} \qquad \forall\, i \in I^1\backslash\{\underline{m}_1, \overline{m}_1 - 1, \overline{m}_1\} \quad (224)$$

$$\tilde{y}^0_r + \tilde{v}^{d,r}_{1,\underline{m}_1} - (\tilde{r}^r_{1,\underline{m}_1} + \tilde{r}^u_{1,\underline{m}_1}) = r^r_{1,\underline{m}_1} + r^u_{1,\underline{m}_1} \quad (225)$$

$$\tilde{r}^r_{1,\overline{m}_1-1} + \tilde{v}^{d,r}_{1,\overline{m}_1-1} - \tilde{r}^u_{1,\overline{m}_1-1} = r^u_{1,\overline{m}_1-1} \quad (226)$$

$$\tilde{r}^r_{q,i-1} + \tilde{v}^{u,r}_{q-1,i} + \tilde{v}^{d,r}_{qi} - (\tilde{r}^r_{qi} + \tilde{r}^u_{qi} + \tilde{r}^d_{qi}) = r^r_{qi} + r^u_{qi} + r^d_{qi}$$
$$\forall\, q \in F\backslash\{1, p+1\}, i \in I^q\backslash\{\underline{m}_{q-1}, \overline{m}_{q-1} - 1, \overline{m}_{q-1}\} \text{ with } \underline{m}_q - 1 < i < \overline{m}_q \quad (227)$$

$$\tilde{r}^r_{q,i-1} + \tilde{v}^{u,r}_{q-1,i} - (\tilde{r}^r_{qi} + \tilde{r}^d_{qi}) = r^r_{qi} + r^d_{qi}$$
$$\forall\, q \in F\backslash\{1, p+1\}, i \in I^q\backslash\{\underline{m}_{q-1}, \overline{m}_{q-1} - 1, \overline{m}_{q-1}\} \text{ with } i < \underline{m}_q - 1 \text{ or } i > \overline{m}_q \quad (228)$$

$$\tilde{r}^r_{q,i-1} + \tilde{v}^{u,r}_{q-1,i} + \tilde{v}^{d,r}_{qi} - (\tilde{r}^r_{qi} + \tilde{r}^d_{qi}) = r^r_{qi} + r^d_{qi} \qquad \forall\, q \in F\backslash\{1, p+1\}, i \in I^q\backslash\{\underline{m}_{q-1}, \overline{m}_{q-1} - 1, \overline{m}_{q-1}\} \text{ with } i = \overline{m}_q \quad (229)$$

$$\tilde{r}^r_{q,i-1} + \tilde{v}^{u,r}_{q-1,i} - (\tilde{r}^r_{qi} + \tilde{r}^u_{qi} + \tilde{r}^d_{qi}) = r^r_{qi} + r^u_{qi} + r^d_{qi}$$
$$\forall\, q \in F\backslash\{1, p+1\}, i \in I^q\backslash\{\underline{m}_{q-1}, \overline{m}_{q-1} - 1, \overline{m}_{q-1}\} \text{ with } i = \underline{m}_q - 1 \quad (230)$$

$$\tilde{v}^{u,r}_{q-1,\underline{m}_{q-1}} + \tilde{v}^{d,r}_{q,\underline{m}_{q-1}} - (\tilde{r}^r_{q,\underline{m}_{q-1}} + \tilde{r}^u_{q,\underline{m}_{q-1}} + \tilde{r}^d_{q,\underline{m}_{q-1}}) = r^r_{q,\underline{m}_{q-1}} + r^u_{q,\underline{m}_{q-1}} + r^d_{q,\underline{m}_{q-1}}$$
$$\forall\, q \in F\backslash\{1, p+1\} \text{ with } \underline{m}_q = \underline{m}_{q-1} \quad (231)$$

$$\tilde{v}^{u,r}_{q-1,\underline{m}_{q-1}} - (\tilde{r}^r_{q,\underline{m}_{q-1}} + \tilde{r}^u_{q,\underline{m}_{q-1}} + \tilde{r}^d_{q,\underline{m}_{q-1}}) = r^r_{q,\underline{m}_{q-1}} + r^u_{q,\underline{m}_{q-1}} + r^d_{q,\underline{m}_{q-1}} \qquad \forall\, q \in F\backslash\{1, p+1\} \text{ with } \underline{m}_q = \underline{m}_{q-1} + 1 \quad (232)$$

$$\tilde{v}^{u,r}_{q-1,\underline{m}_{q-1}} - (\tilde{r}^r_{q,\underline{m}_{q-1}} + \tilde{r}^d_{q,\underline{m}_{q-1}}) = r^r_{q,\underline{m}_{q-1}} + r^d_{q,\underline{m}_{q-1}} \qquad \forall\, q \in F\backslash\{1, p+1\} \text{ with } \underline{m}_q > \underline{m}_{q-1} + 1 \quad (233)$$

$$\tilde{r}^r_{q,\overline{m}_{q-1}-2} + \tilde{v}^{u,r}_{q-1,\overline{m}_{q-1}-1} + \tilde{v}^{d,r}_{q,\overline{m}_{q-1}-1} - (\tilde{r}^u_{q,\overline{m}_{q-1}-1} + \tilde{r}^d_{q,\overline{m}_{q-1}-1}) = r^u_{q,\overline{m}_{q-1}-1} + r^d_{q,\overline{m}_{q-1}-1}$$
$$\forall\, q \in F\backslash\{1, p+1\} \text{ with } \overline{m}_q = \overline{m}_{q-1} \quad (234)$$

$$\tilde{r}^r_{q,\overline{m}_{q-1}-2} + \tilde{v}^{u,r}_{q-1,\overline{m}_{q-1}-1} + \tilde{v}^{d,r}_{q,\overline{m}_{q-1}-1} - \tilde{r}^d_{q,\overline{m}_{q-1}-1} = r^d_{q,\overline{m}_{q-1}-1} \qquad \forall\, q \in F\backslash\{1, p+1\} \text{ with } \overline{m}_q = \overline{m}_{q-1} - 1 \quad (235)$$

$$\tilde{r}^r_{q,\overline{m}_{q-1}-2} + \tilde{v}^{u,r}_{q-1,\overline{m}_{q-1}-1} - \tilde{r}^d_{q,\overline{m}_{q-1}-1} = r^d_{q,\overline{m}_{q-1}-1} \qquad \forall\, q \in F\setminus\{1,p+1\} \text{ with } \overline{m}_q < \overline{m}_{q-1}-1 \quad (236)$$

$$\tilde{r}^r_{p+1,i-1} + \tilde{v}^{u,r}_{pi} - (\tilde{r}^r_{p+1,i} + \tilde{r}^d_{p+1,i}) = r^r_{p+1,i} + r^d_{p+1,i} \qquad \forall\, i \in I^p\setminus\{\underline{m}_p, \overline{m}_p-1, \overline{m}_p\} \quad (237)$$

$$\tilde{v}^{u,r}_{p,\underline{m}_p} - (\tilde{r}^r_{p+1,\underline{m}_p} + \tilde{r}^d_{p+1,\underline{m}_p}) = r^r_{p+1,\underline{m}_p} + r^d_{p+1,\underline{m}_p} \quad (238)$$

$$\tilde{r}^r_{p+1,\overline{m}_p-2} + \tilde{v}^{u,r}_{p,\overline{m}_p-1} - \tilde{r}^d_{p+1,\overline{m}_p-1} = r^d_{p+1,\overline{m}_p-1} \quad (239)$$

- Constraints corresponding to vertices $[l,q,i]$

$$\tilde{l}^l_{1,i+1} + \tilde{v}^{d,l}_{1i} - (\tilde{l}^l_{1i} + \tilde{l}^u_{1i}) = l^l_{1i} + l^u_{1i} \qquad \forall\, i \in I^1\setminus\{\underline{m}_1, \overline{m}_1\} \quad (240)$$

$$\tilde{y}^0_l + \tilde{l}^l_{1,\underline{m}_1+1} + \tilde{v}^{d,l}_{1,\underline{m}_1} - \tilde{y}^l_0 = y^l_0 \quad (241)$$

$$\tilde{v}^{d,l}_{1,\overline{m}_1} - (\tilde{l}^l_{1,\overline{m}_l} + \tilde{l}^u_{1,\overline{m}_l}) = l^l_{1,\overline{m}_l} + l^u_{1,\overline{m}_l} \quad (242)$$

$$\tilde{l}^l_{q,i+1} + \tilde{v}^{d,l}_{q,i} + \tilde{v}^{u,l}_{q-1,i} - (\tilde{l}^l_{qi} + \tilde{l}^u_{qi} + \tilde{l}^d_{qi}) = l^l_{qi} + l^u_{qi} + l^d_{qi}$$
$$\forall\, q \in F\setminus\{1,p+1\}, i \in I^q\setminus\{\underline{m}_{q-1}, \underline{m}_{q-1}+1, \overline{m}_{q-1}\} \text{ with } \underline{m}_q < i < \overline{m}_q+1 \quad (243)$$

$$\tilde{l}^l_{q,i+1} + \tilde{v}^{u,l}_{q-1,i} - (\tilde{l}^l_{qi} + \tilde{l}^d_{qi}) = l^l_{qi} + l^d_{qi}$$
$$\forall\, q \in F\setminus\{1,p+1\}, i \in I^q\setminus\{\underline{m}_{q-1}, \underline{m}_{q-1}+1, \overline{m}_{q-1}\} \text{ with } i < \underline{m}_q \text{ or } i > \overline{m}_q+1 \quad (244)$$

$$\tilde{l}^l_{q,i+1} + \tilde{v}^{d,l}_{q,i} + \tilde{v}^{u,l}_{q-1,i} - (\tilde{l}^l_{qi} + \tilde{l}^d_{qi}) = l^l_{qi} + l^d_{qi} \qquad \forall\, q \in F\setminus\{1,p+1\}, i \in I^q\setminus\{\underline{m}_{q-1}, \underline{m}_{q-1}+1, \overline{m}_{q-1}\} \text{ with } i = \underline{m}_q \quad (245)$$

$$\tilde{l}^l_{q,i+1} + \tilde{v}^{u,l}_{q-1,i} - (\tilde{l}^l_{qi} + \tilde{l}^u_{qi} + \tilde{l}^d_{qi}) = l^l_{qi} + l^u_{qi} + l^d_{qi}$$
$$\forall\, q \in F\setminus\{1,p+1\}, i \in I^q\setminus\{\underline{m}_{q-1}, \underline{m}_{q-1}+1, \overline{m}_{q-1}\} \text{ with } i = \overline{m}_q+1 \quad (246)$$

$$\tilde{l}^l_{q,\underline{m}_{q-1}+2} + \tilde{v}^{d,l}_{q,\underline{m}_{q-1}+1} + \tilde{v}^{u,l}_{q-1,\underline{m}_{q-1}+1} - (\tilde{l}^u_{q,\underline{m}_{q-1}+1} + \tilde{l}^d_{q,\underline{m}_{q-1}+1}) = l^u_{q,\underline{m}_{q-1}+1} + l^d_{q,\underline{m}_{q-1}+1}$$
$$\forall\, q \in F\setminus\{1,p+1\} \text{ with } \underline{m}_q = \underline{m}_{q-1} \quad (247)$$

$$\tilde{l}^l_{q,\underline{m}_{q-1}+2} + \tilde{v}^{d,l}_{q,\underline{m}_{q-1}+1} + \tilde{v}^{u,l}_{q-1,\underline{m}_{q-1}+1} - \tilde{l}^d_{q,\underline{m}_{q-1}+1} = l^d_{q,\underline{m}_{q-1}+1} \qquad \forall\, q \in F\setminus\{1,p+1\} \text{ with } \underline{m}_q = \underline{m}_{q-1}+1 \quad (248)$$

$$\tilde{l}^l_{q,\underline{m}_{q-1}+2} + \tilde{v}^{u,l}_{q-1,\underline{m}_{q-1}+1} - \tilde{l}^d_{q,\underline{m}_{q-1}+1} = l^d_{q,\underline{m}_{q-1}+1} \qquad \forall\, q \in F\setminus\{1,p+1\} \text{ with } \underline{m}_q > \underline{m}_{q-1}+1 \quad (249)$$

$$\tilde{v}^{d,l}_{q,\overline{m}_{q-1}} + \tilde{v}^{u,l}_{q-1,\overline{m}_{q-1}} - (\tilde{l}^l_{q,\overline{m}_{q-1}} + \tilde{l}^u_{q,\overline{m}_{q-1}} + \tilde{l}^d_{q,\overline{m}_{q-1}}) = l^l_{q,\overline{m}_{q-1}} + l^u_{q,\overline{m}_{q-1}} + l^d_{q,\overline{m}_{q-1}}$$
$$\forall\, q \in F\setminus\{1,p+1\} \text{ with } \overline{m}_q = \overline{m}_{q-1} \quad (250)$$

$$\tilde{v}^{u,l}_{q-1,\overline{m}_{q-1}} - (\tilde{l}^l_{q,\overline{m}_{q-1}} + \tilde{l}^u_{q,\overline{m}_{q-1}} + \tilde{l}^d_{q,\overline{m}_{q-1}}) = l^l_{q,\overline{m}_{q-1}} + l^u_{q,\overline{m}_{q-1}} + l^d_{q,\overline{m}_{q-1}} \qquad \forall\, q \in F\setminus\{1,p+1\} \text{ with } \overline{m}_q = \overline{m}_{q-1}-1 \quad (251)$$

$$\tilde{v}^{u,l}_{q-1,\overline{m}_{q-1}} - (\tilde{l}^l_{q,\overline{m}_{q-1}} + \tilde{l}^d_{q,\overline{m}_{q-1}}) = l^l_{q,\overline{m}_{q-1}} + l^d_{q,\overline{m}_{q-1}} \qquad \forall\, q \in F\setminus\{1,p+1\} \text{ with } \overline{m}_q < \overline{m}_{q-1}-1 \quad (252)$$

$$\tilde{l}^l_{p+1,i+1} + \tilde{v}^{u,l}_{pi} - (\tilde{l}^l_{p+1,i} + \tilde{l}^d_{p+1,i}) = l^l_{p+1,i} + l^d_{p+1,i} \qquad \forall\, i \in I^p\setminus\{\underline{m}_p, \underline{m}_p+1, \overline{m}_p\} \quad (253)$$

$$\tilde{l}^l_{p+1,\underline{m}_p+2} + \tilde{v}^{u,l}_{p,\underline{m}_p+1} - \tilde{l}^d_{p+1,\underline{m}_p+1} = l^d_{p+1,\underline{m}_p+1} \quad (254)$$

$$\tilde{v}^{u,l}_{p,\overline{m}_p} - (\tilde{l}^l_{p+1,\overline{m}_p} + \tilde{l}^d_{p+1,\overline{m}_p}) = l^l_{p+1,\overline{m}_p} + l^d_{p+1,\overline{m}_p} \quad (255)$$

- Constraints corresponding to vertices $[u,q,i]$

$$\tilde{l}^u_{q,i+1} + \tilde{r}^u_{q,i-1} + \tilde{v}^u_{q-1,i} - (\tilde{v}^u_{qi} + \tilde{v}^{u,l}_{qi} + \tilde{v}^{u,r}_{qi}) = v^u_{qi} + v^{u,l}_{qi} + v^{u,r}_{qi} \quad (256)$$
$$\forall\, q \in B\setminus\{1,p\}, i \in I^q_1\setminus\{\underline{m}_q, \overline{m}_q\} \text{ with } i \geq \underline{m}_{q+1}, i \leq \overline{m}_{q+1} \quad (257)$$

$$\tilde{l}^u_{1,\underline{m}_1+1} + \tilde{y}^0_u - (\tilde{v}^u_{1,\underline{m}_1} + \tilde{v}^{u,r}_{1,\underline{m}_1}) = v^u_{1,\underline{m}_1} + v^{u,r}_{1,\underline{m}_1} \qquad \text{if } \underline{m}_1 = \underline{m}_2 \text{ and } \underline{m}_1 \in I^1_1 \quad (258)$$

$$\tilde{r}^u_{q,\overline{m}_q-1} + \tilde{v}^u_{q-1,\overline{m}_q} - (\tilde{v}^u_{q,\overline{m}_q} + \tilde{v}^{u,l}_{q,\overline{m}_q}) = v^u_{q,\overline{m}_q} + v^{u,l}_{q,\overline{m}_q}$$

$$\forall\ q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^q_1 \text{ and } \overline{m}_q = \overline{m}_{q+1} \text{ and } \overline{m}_q = \overline{m}_{q-1} \quad (259)$$

$$\tilde{r}^u_{q,\overline{m}_q-1} + \tilde{l}^u_{q,\overline{m}_q+1} + \tilde{v}^u_{q-1,\overline{m}_q} - (\tilde{v}^u_{q,\overline{m}_q} + \tilde{v}^{u,l}_{q,\overline{m}_q}) = v^u_{q,\overline{m}_q} + v^{u,l}_{q,\overline{m}_q}$$

$$\forall\ q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^q_1 \text{ and } \overline{m}_q = \overline{m}_{q+1} \text{ and } \overline{m}_q \neq \overline{m}_{q-1} \quad (260)$$

$$\tilde{l}^u_{q,i+1} + \tilde{r}^u_{q,i-1} + \tilde{v}^u_{q-1,i} - (\tilde{v}^{u,l}_{qi} + \tilde{v}^{u,r}_{qi}) = v^{u,l}_{q,} + v^{u,r}_{qi}$$

$$\forall\ q \in B\backslash\{1,p\}, i \in I^q_1\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \quad (261)$$

$$\tilde{l}^u_{1,\underline{m}_1+1} + \tilde{y}^0_u - \tilde{v}^{u,r}_{1,\underline{m}_1} = v^{u,r}_{1,\underline{m}_1} \qquad \text{if } \underline{m}_1 \neq \underline{m}_2 \text{ and } \underline{m}_1 \in I^1_1 \quad (262)$$

$$\tilde{r}^u_{q,\overline{m}_q-1} + \tilde{v}^u_{q-1,\overline{m}_q} - \tilde{v}^{u,l}_{q,\overline{m}_q} = v^{u,l}_{q,\overline{m}_q} \qquad \forall\ q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^q_1 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \text{ and } \overline{m}_q = \overline{m}_{q-1} \quad (263)$$

$$\tilde{r}^u_{q,\overline{m}_q-1} + \tilde{l}^u_{q,\overline{m}_q+1} + \tilde{v}^u_{q-1,\overline{m}_q} - \tilde{v}^{u,l}_{q,\overline{m}_q} = v^{u,l}_{q,\overline{m}_q} \qquad \forall\ q \in B\backslash\{1,p\} \text{ with } \overline{m}_q \in I^q_1 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \text{ and } \overline{m}_q \neq \overline{m}_{q-1} \quad (264)$$

$$\tilde{l}^u_{q,\underline{m}_q+1} + \tilde{v}^u_{q-1,\underline{m}_q} - (\tilde{v}^u_{q,\underline{m}_q} + \tilde{v}^{u,r}_{q,\underline{m}_q}) = v^u_{q,\underline{m}_q} + v^{u,r}_{q,\underline{m}_q} \quad (265)$$

$$\forall\ q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \in I^q_1 \text{ and } \underline{m}_q = \underline{m}_{q+1} \text{ and } \underline{m}_q = \underline{m}_{q-1} \quad (266)$$

$$\tilde{r}^u_{q,\underline{m}_q-1} + \tilde{l}^u_{q,\underline{m}_q+1} + \tilde{v}^u_{q-1,\underline{m}_q} - (\tilde{v}^u_{q,\underline{m}_q} + \tilde{v}^{u,r}_{q,\underline{m}_q}) = v^u_{q,\underline{m}_q} + v^{u,r}_{q,\underline{m}_q} \quad (267)$$

$$\forall\ q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \in I^q_1 \text{ and } \underline{m}_q = \underline{m}_{q+1} \text{ and } \underline{m}_q \neq \underline{m}_{q-1} \quad (268)$$

$$\tilde{l}^u_{q,\underline{m}_q+1} + \tilde{v}^u_{q-1,\underline{m}_q} - \tilde{v}^{u,r}_{q,\underline{m}_q} = v^{u,r}_{q,\underline{m}_q} \qquad \forall\ q \in B\backslash\{1,p\} \text{ with } \underline{m}_q \neq \underline{m}_{q+1} \text{ and } \underline{m}_q \in I^q_1 \quad (269)$$

$$\tilde{l}^u_{p,i+1} + \tilde{r}^u_{p,i-1} + \tilde{v}^u_{p-1,i} - (\tilde{v}^{u,l}_{pi} + \tilde{v}^{u,r}_{pi}) = v^{u,l}_{pi} + v^{u,r}_{pi} \qquad \forall\ i \in I^p_1\backslash\{\underline{m}_p, \overline{m}_p\} \quad (270)$$

$$\tilde{l}^u_{p,\underline{m}_p+1} + \tilde{v}^u_{p-1,\underline{m}_p} - \tilde{v}^{u,r}_{p,\underline{m}_p} = v^{u,r}_{p,\underline{m}_p} \qquad \text{if } \underline{m}_p \in I^p_1 \quad (271)$$

$$\tilde{r}^u_{p,\overline{m}_p-1} + \tilde{v}^u_{p-1,\overline{m}_p} - \tilde{v}^{u,l}_{p,\overline{m}_p} = v^{u,l}_{p,\overline{m}_p} \qquad \text{if } \overline{m}_p \in I^p_1 \quad (272)$$

$$\tilde{l}^u_{1,i+1} + \tilde{r}^u_{1,i-1} - (\tilde{v}^u_{1i} + \tilde{v}^{u,l}_{1i} + \tilde{v}^{u,r}_{1i}) = v^u_{1i} + v^{u,l}_{1i} + v^{u,r}_{1i} \qquad \forall\ i \in I^1_1\backslash\{\underline{m}_1, \overline{m}_1\} \text{ with } \underline{m}_2 \leq i \leq \overline{m}_2 \quad (273)$$

$$\tilde{l}^u_{1,i+1} + \tilde{r}^u_{1,i-1} - (\tilde{v}^{u,l}_{1i} + \tilde{v}^{u,r}_{1i}) = v^{u,l}_{1i} + v^{u,r}_{1i} \qquad \forall\ i \in I^1_1\backslash\{\underline{m}_1, \overline{m}_1\} \text{ with } i < \underline{m}_2 \text{ or } i > \overline{m}_2 \quad (274)$$

$$\tilde{r}^u_{1,\overline{m}_1-1} - (\tilde{v}^u_{1,\overline{m}_1} + \tilde{v}^{u,l}_{1,\overline{m}_1}) = v^u_{1,\overline{m}_1} + v^{u,l}_{1,\overline{m}_1} \qquad \text{if } \overline{m}_1 = \overline{m}_2 \text{ and } \overline{m}_1 \in I^1_1 \quad (275)$$

$$\tilde{r}^u_{1,\overline{m}_1-1} - \tilde{v}^{u,l}_{1,\overline{m}_1} = v^{u,l}_{1,\overline{m}_1} \qquad \text{if } \overline{m}_1 \neq \overline{m}_2 \text{ and } \overline{m}_1 \in I^1_1 \quad (276)$$

$$\tilde{l}^u_{1,i+1} + \tilde{r}^u_{1,i-1} - (\tilde{t}^u_{1i} + \tilde{e}^d_{1i1}) = t^u_{1i} + e^d_{1i1} \qquad \forall i \in I^1_2\backslash\{\underline{m}_1, \overline{m}_1\} \quad (277)$$

$$\tilde{l}^u_{1,\underline{m}_1+1} + \tilde{y}^0_u - (\tilde{t}^u_{1,\underline{m}_1} + \tilde{e}^d_{1,\underline{m}_1,1}) = t^u_{1,\underline{m}_1} + e^d_{1,\underline{m}_1,1} \qquad \text{if } \underline{m}_1 \in I^1_2 \quad (278)$$

$$\tilde{r}^u_{1,\overline{m}_1-1} - (\tilde{t}^u_{1,\overline{m}_1} + \tilde{e}^d_{1,\overline{m}_1,1}) = t^u_{1,\overline{m}_1} + e^d_{1,\overline{m}_1,1} \qquad \text{if } \overline{m}_1 \in I^1_2 \quad (279)$$

$$\tilde{l}^u_{q,i+1} + \tilde{r}^u_{q,i-1} + \tilde{v}^u_{q-1,i} - (\tilde{t}^u_{qi} + \tilde{e}^d_{qi1}) = t^u_{qi} + e^d_{qi1} \qquad \forall\ q \in B\backslash\{1\}, i \in I^q_2\backslash\{\underline{m}_q, \overline{m}_q\} \quad (280)$$

$$\tilde{l}^u_{q,\underline{m}_q+1} + \tilde{v}^u_{q-1,\underline{m}_q} - (\tilde{t}^u_{q,\underline{m}_q} + \tilde{e}^d_{q,\underline{m}_q,1}) = t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_2 \text{ and } \underline{m}_q = \underline{m}_{q-1} \quad (281)$$

$$\tilde{r}^u_{q,\underline{m}_q-1} + \tilde{l}^u_{q,\underline{m}_q+1} + \tilde{v}^u_{q-1,\underline{m}_q} - (\tilde{t}^u_{q,\underline{m}_q} + \tilde{e}^d_{q,\underline{m}_q,1}) = t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_2 \text{ and } \underline{m}_q \neq \underline{m}_{q-1} \quad (282)$$

$$\tilde{r}^u_{q,\overline{m}_q-1} + \tilde{v}^u_{q-1,\overline{m}_q} - (\tilde{t}^u_{q,\overline{m}_q} + \tilde{e}^d_{q,\overline{m}_q,1}) = t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_2 \text{ and } \overline{m}_q = \overline{m}_{q-1} \quad (283)$$

$$\tilde{r}^u_{q,\overline{m}_q-1} + \tilde{l}^u_{q,\overline{m}_q+1} + \tilde{v}^u_{q-1,\overline{m}_q} - (\tilde{t}^u_{q,\overline{m}_q} + \tilde{e}^d_{q,\overline{m}_q,1}) = t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_2 \text{ and } \overline{m}_q \neq \overline{m}_{q-1} \quad (284)$$

$$\tilde{l}^u_{1,i+1} + \tilde{r}^u_{1,i-1} - (\tilde{t}^u_{1i} + \tilde{e}^d_{1i1} + \tilde{e}^d_{1i3}) = t^u_{1i} + e^d_{1i1} + e^d_{1i3} \qquad \forall\ i \in (I^1_0 \cup I^1_4)\backslash\{\underline{m}_1, \overline{m}_1\} \quad (285)$$

$$\tilde{l}^u_{1,\underline{m}_1+1} + \tilde{y}^0_u - (\tilde{t}^u_{1,\underline{m}_1} + \tilde{e}^d_{1,\underline{m}_1,1} + \tilde{e}^d_{1,\underline{m}_1,3}) = t^u_{1,\underline{m}_1} + e^d_{1,\underline{m}_1,1} + e^d_{1,\underline{m}_1,3} \qquad \text{if } \underline{m}_1 \in I^1_0 \cup I^1_4 \quad (286)$$

$$\tilde{r}^u_{1,\overline{m}_1-1} - (\tilde{t}^u_{1,\overline{m}_1} + \tilde{e}^d_{1,\overline{m}_1,1} + \tilde{e}^d_{1,\overline{m}_1,3}) = t^u_{1,\overline{m}_1} + e^d_{1,\overline{m}_1,1} + e^d_{1,\overline{m}_1,3} \qquad \text{if } \overline{m}_1 \in I^1_0 \cup I^1_4 \quad (287)$$

$$\tilde{l}^u_{q,i+1} + \tilde{r}^u_{q,i-1} + \tilde{v}^u_{q-1,i} - (\tilde{t}^u_{qi} + \tilde{e}^d_{qi1} + \tilde{e}^d_{qi3}) = t^u_{qi} + e^d_{qi1} + e^d_{qi3} \qquad \forall\ q \in B\backslash\{1\}, i \in (I^q_0 \cup I^q_4)\backslash\{\underline{m}_q, \overline{m}_q\} \quad (288)$$

$$\tilde{l}^u_{q,\underline{m}_q+1} + \tilde{v}^u_{q-1,\underline{m}_q} - (\tilde{t}^u_{q,\underline{m}_q} + \tilde{e}^d_{q,\underline{m}_q,1} + \tilde{e}^d_{q,\underline{m}_q,3}) = t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} + e^d_{q,\underline{m}_q,3}$$
$$\forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_0 \cup I^q_4 \text{ and } \underline{m}_q = \underline{m}_{q-1} \quad (289)$$

$$\tilde{r}^u_{q,\underline{m}_q-1} + \tilde{l}^u_{q,\underline{m}_q+1} + \tilde{v}^u_{q-1,\underline{m}_q} - (\tilde{t}^u_{q,\underline{m}_q} + \tilde{e}^d_{q,\underline{m}_q,1} + \tilde{e}^d_{q,\underline{m}_q,3}) = t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} + e^d_{q,\underline{m}_q,3}$$
$$\forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_0 \cup I^q_4 \text{ and } \underline{m}_q \neq \underline{m}_{q-1} \quad (290)$$

$$\tilde{r}^u_{q,\overline{m}_q-1} + \tilde{v}^u_{q-1,\overline{m}_q} - (\tilde{t}^u_{q,\overline{m}_q} + \tilde{e}^d_{q,\overline{m}_q,1} + \tilde{e}^d_{q,\overline{m}_q,3}) = t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} + e^d_{q,\overline{m}_q,3}$$
$$\forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_0 \cup I^q_4 \text{ and } \overline{m}_q = \overline{m}_{q-1} \quad (291)$$

$$\tilde{r}^u_{q,\overline{m}_q-1} + \tilde{l}^u_{q,\overline{m}_q+1} + \tilde{v}^u_{q-1,\overline{m}_q} - (\tilde{t}^u_{q,\overline{m}_q} + \tilde{e}^d_{q,\overline{m}_q,1} + \tilde{e}^d_{q,\overline{m}_q,3}) = t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} + e^d_{q,\overline{m}_q,3}$$
$$\forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_0 \cup I^q_4 \text{ and } \overline{m}_q \neq \overline{m}_{q-1} \quad (292)$$

$$\tilde{l}^u_{1,i+1} + \tilde{r}^u_{1,i-1} - (\tilde{t}^u_{1i} + \tilde{e}^d_{1i1} + \tilde{e}^d_{1i2}) = t^u_{1i} + e^d_{1i1} + e^d_{1i2} \qquad \forall\ i \in (I^1_3 \cup I^1_5)\backslash\{\underline{m}_1, \overline{m}_1\} \quad (293)$$

$$\tilde{l}^u_{1,\underline{m}_1+1} + \tilde{y}^0_u - (\tilde{t}^u_{1,\underline{m}_1} + \tilde{e}^d_{1,\underline{m}_1,1} + \tilde{e}^d_{1,\underline{m}_1,2}) = t^u_{1,\underline{m}_1} + e^d_{1,\underline{m}_1,1} + e^d_{1,\underline{m}_1,2} \qquad \text{if } \underline{m}_1 \in I^1_3 \cup I^1_5 \quad (294)$$

$$\tilde{r}^u_{1,\overline{m}_1-1} - (\tilde{t}^u_{1,\overline{m}_1} + \tilde{e}^d_{1,\overline{m}_1,1} + \tilde{e}^d_{1,\overline{m}_1,2}) = t^u_{1,\overline{m}_1} + e^d_{1,\overline{m}_1,1} + e^d_{1,\overline{m}_1,2} \qquad \text{if } \overline{m}_1 \in I^1_3 \cup I^1_5 \quad (295)$$

$$\tilde{l}^u_{q,i+1} + \tilde{r}^u_{q,i-1} + \tilde{v}^u_{q-1,i} - (\tilde{t}^u_{qi} + \tilde{e}^d_{qi1} + \tilde{e}^d_{qi2}) = t^u_{qi} + e^d_{qi1} + e^d_{qi2} \qquad \forall\ q \in B\backslash\{1\}, i \in (I^q_3 \cup I^q_5)\backslash\{\underline{m}_q, \overline{m}_q\} \quad (296)$$

$$\tilde{l}^u_{q,\underline{m}_q+1} + \tilde{v}^u_{q-1,\underline{m}_q} - (\tilde{t}^u_{q,\underline{m}_q} + \tilde{e}^d_{q,\underline{m}_q,1} + \tilde{e}^d_{q,\underline{m}_q,2}) = t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} + e^d_{q,\underline{m}_q,2}$$
$$\forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_3 \cup I^q_5 \text{ and } \underline{m}_q = \underline{m}_{q-1} \quad (297)$$

$$\tilde{r}^u_{q,\underline{m}_q-1} + \tilde{l}^u_{q,\underline{m}_q+1} + \tilde{v}^u_{q-1,\underline{m}_q} - (\tilde{t}^u_{q,\underline{m}_q} + \tilde{e}^d_{q,\underline{m}_q,1} + \tilde{e}^d_{q,\underline{m}_q,2}) = t^u_{q,\underline{m}_q} + e^d_{q,\underline{m}_q,1} + e^d_{q,\underline{m}_q,2}$$
$$\forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_3 \cup I^q_5 \text{ and } \underline{m}_q \neq \underline{m}_{q-1} \quad (298)$$

$$\tilde{r}^u_{q,\overline{m}_q-1} + \tilde{v}^u_{q-1,\overline{m}_q} - (\tilde{t}^u_{q,\overline{m}_q} + \tilde{e}^d_{q,\overline{m}_q,1} + \tilde{e}^d_{q,\overline{m}_q,2}) = t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} + e^d_{q,\overline{m}_q,2}$$
$$\forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_3 \cup I^q_5 \text{ and } \overline{m}_q = \overline{m}_{q-1} \quad (299)$$

$$\tilde{r}^u_{q,\overline{m}_q-1} + \tilde{l}^u_{q,\overline{m}_q+1} + \tilde{v}^u_{q-1,\overline{m}_q} - (\tilde{t}^u_{q,\overline{m}_q} + \tilde{e}^d_{q,\overline{m}_q,1} + \tilde{e}^d_{q,\overline{m}_q,2}) = t^u_{q,\overline{m}_q} + e^d_{q,\overline{m}_q,1} + e^d_{q,\overline{m}_q,2}$$
$$\forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_3 \cup I^q_5 \text{ and } \overline{m}_q \neq \overline{m}_{q-1} \quad (300)$$

- Constraints corresponding to vertices $[d,q,i]$

$$\tilde{l}^d_{2,i+1} + \tilde{r}^d_{2,i-1} + \tilde{v}^d_{2i} - (\tilde{v}^{d,l}_{1i} + \tilde{v}^{d,r}_{1i}) = v^{d,l}_{1i} + v^{d,r}_{1i} \qquad \forall\ i \in I^1_1\backslash\{\underline{m}_1, \overline{m}_1\} \text{ with } \underline{m}_2 \leq i \leq \overline{m}_2 \quad (301)$$

$$\tilde{l}^d_{2,i+1} + \tilde{r}^d_{2,i-1} - (\tilde{v}^{d,l}_{1i} + \tilde{v}^{d,r}_{1i}) = v^{d,l}_{1i} + v^{d,r}_{1i} \qquad \forall\ i \in I^1_1\backslash\{\underline{m}_1, \overline{m}_1\} \text{ with } i < \underline{m}_2 \text{ or } i > \overline{m}_2 \quad (302)$$

$$\tilde{l}^d_{2,\underline{m}_1+1} + \tilde{v}^d_{2,\underline{m}_1} - (\tilde{v}^{d,l}_{1,\underline{m}_1} + \tilde{v}^{d,r}_{1,\underline{m}_1}) = v^{d,l}_{1,\underline{m}_1} + v^{d,r}_{1,\underline{m}_1} \qquad \text{if } \underline{m}_1 \in I^1_1 \text{ and } \underline{m}_1 = \underline{m}_2 \quad (303)$$

$$\tilde{l}^d_{2,\underline{m}_1+1} - (\tilde{v}^{d,l}_{1,\underline{m}_1} + \tilde{v}^{d,r}_{1,\underline{m}_1}) = v^{d,l}_{1,\underline{m}_1} + v^{d,r}_{1,\underline{m}_1} \qquad \text{if } \underline{m}_1 \in I^1_1 \text{ and } \underline{m}_1 \neq \underline{m}_2 \quad (304)$$

$$\tilde{r}^d_{2,\overline{m}_1-1} + \tilde{v}^d_{2,\overline{m}_1} - \tilde{v}^{d,l}_{1,\overline{m}_1} = v^{d,l}_{1,\overline{m}_1} \qquad \text{if } \overline{m}_1 \in I^1_1 \text{ and } \overline{m}_1 = \overline{m}_2 \quad (305)$$

$$\tilde{r}^d_{2,\overline{m}_1-1} - \tilde{v}^{d,l}_{1,\overline{m}_1} = v^{d,l}_{1,\overline{m}_1} \qquad \text{if } \overline{m}_1 \in I^1_1 \text{ and } \overline{m}_1 \neq \overline{m}_2 \quad (306)$$

$$\tilde{l}^d_{q+1,i+1} + \tilde{r}^d_{q+1,i-1} + \tilde{v}^d_{q+1,i} - (\tilde{v}^{d,l}_{qi} + \tilde{v}^{d,r}_{qi} + \tilde{v}^d_{qi}) = v^{d,l}_{qi} + v^{d,r}_{qi} + v^d_{qi}$$
$$\forall\ q \in B\backslash\{1,p\}, i \in I^q_1\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } \underline{m}_{q+1} \leq i \leq \overline{m}_{q+1} \quad (307)$$

$$\tilde{l}^d_{q+1,i+1} + \tilde{r}^d_{q+1,i-1} - (\tilde{v}^{d,l}_{qi} + \tilde{v}^{d,r}_{qi} + \tilde{v}^d_{qi}) = v^{d,l}_{qi} + v^{d,r}_{qi} + v^d_{qi}$$
$$\forall\ q \in B\backslash\{1,p\}, i \in I^q_1\backslash\{\underline{m}_q, \overline{m}_q\}, \text{ with } i < \underline{m}_{q+1} \text{ or } 1 > \overline{m}_{q+1} \quad (308)$$

$$\tilde{l}^d_{q+1,\underline{m}_q+1}+\tilde{v}^d_{q+1,\underline{m}_q}-(\tilde{v}^{d,r}_{q,\underline{m}_q}+\tilde{v}^d_{q,\underline{m}_q})=v^{d,r}_{q,\underline{m}_q}+v^d_{q,\underline{m}_q} \quad \forall\, q\in B\backslash\{1,p\} \text{ with } \underline{m}_q\in I^q_1 \text{ and } \underline{m}_q=\underline{m}_{q+1},\underline{m}_q=\underline{m}_{q-1} \tag{309}$$

$$\tilde{l}^d_{q+1,\underline{m}_q+1}-(\tilde{v}^{d,r}_{q,\underline{m}_q}+\tilde{v}^d_{q,\underline{m}_q})=v^{d,r}_{q,\underline{m}_q}+v^d_{q,\underline{m}_q} \quad \forall\, q\in B\backslash\{1,p\} \text{ with } \underline{m}_q\in I^q_1 \text{ and } \underline{m}_q\neq\underline{m}_{q+1},\underline{m}_q=\underline{m}_{q-1} \tag{310}$$

$$\tilde{r}^d_{q+1,\overline{m}_q-1}+\tilde{v}^d_{q+1,\overline{m}_q}-(\tilde{v}^{d,l}_{q,\overline{m}_q}+\tilde{v}^d_{q,\overline{m}_q})=v^{d,l}_{q,\overline{m}_q}+v^d_{q,\overline{m}_q} \quad \forall\, q\in B\backslash\{1,p\} \text{ with } \overline{m}_q\in I^q_1 \text{ and } \overline{m}_q=\overline{m}_{q+1},\overline{m}_q=\overline{m}_{q-1} \tag{311}$$

$$\tilde{r}^d_{q+1,\overline{m}_q-1}-(\tilde{v}^{d,l}_{q,\overline{m}_q}+\tilde{v}^d_{q,\overline{m}_q})=v^{d,l}_{q,\overline{m}_q}+v^d_{q,\overline{m}_q} \quad \forall\, q\in B\backslash\{1,p\} \text{ with } \overline{m}_q\in I^q_1 \text{ and } \overline{m}_q\neq\overline{m}_{q+1},\overline{m}_q=\overline{m}_{q-1} \tag{312}$$

$$\tilde{l}^d_{p+1,i+1}+\tilde{r}^d_{p+1,i-1}-(\tilde{v}^{d,l}_{pi}+\tilde{v}^{d,r}_{pi}+\tilde{v}^d_{pi})=v^{d,l}_{pi}+v^{d,r}_{pi}+v^d_{pi} \quad \forall\, i\in I^p_1\backslash\{\underline{m}_p,\overline{m}_p\} \tag{313}$$

$$\tilde{l}^d_{p+1,\underline{m}_p+1}-(\tilde{v}^{d,l}_{p,\underline{m}_p}+\tilde{v}^{d,r}_{p,\underline{m}_p}+\tilde{v}^d_{p,\underline{m}_p})=v^{d,l}_{p,\underline{m}_p}+v^{d,r}_{p,\underline{m}_p}+v^d_{p,\underline{m}_p} \quad \text{if } \underline{m}_p\in I^p_1 \text{ and } \underline{m}_p\neq\underline{m}_{p-1} \tag{314}$$

$$\tilde{l}^d_{p+1,\underline{m}_p+1}-(\tilde{v}^{d,r}_{p,\underline{m}_p}+\tilde{v}^d_{p,\underline{m}_p})=v^{d,r}_{p,\underline{m}_p}+v^d_{p,\underline{m}_p} \quad \text{if } \underline{m}_p\in I^p_1 \text{ and } \underline{m}_p=\underline{m}_{p-1} \tag{315}$$

$$\tilde{r}^d_{p+1,\overline{m}_p-1}-(\tilde{v}^{d,l}_{p,\overline{m}_p}+\tilde{v}^{d,r}_{p,\overline{m}_p}+\tilde{v}^d_{p,\overline{m}_p})=v^{d,l}_{p,\overline{m}_p}+v^{d,r}_{p,\overline{m}_p}+v^d_{p,\overline{m}_p} \quad \text{if } \overline{m}_p\in I^p_1 \text{ and } \overline{m}_p\neq\overline{m}_{p-1} \tag{316}$$

$$\tilde{r}^d_{p+1,\overline{m}_p-1}-(\tilde{v}^{d,l}_{p,\overline{m}_p}+\tilde{v}^d_{p,\overline{m}_p})=v^{d,l}_{p,\overline{m}_p}+v^d_{p,\overline{m}_p} \quad \text{if } \overline{m}_p\in I^p_1 \text{ and } \overline{m}_p=\overline{m}_{p-1} \tag{317}$$

$$\tilde{l}^d_{q+1,\underline{m}_q+1}+\tilde{v}^d_{q+1,\underline{m}_q}-(\tilde{v}^{d,l}_{q,\underline{m}_q}+\tilde{v}^{d,r}_{q,\underline{m}_q}+\tilde{v}^d_{q,\underline{m}_q})=v^{d,l}_{q,\underline{m}_q}+v^{d,r}_{q,\underline{m}_q}+v^d_{q,\underline{m}_q}$$
$$\forall\, q\in B\backslash\{1,p\} \text{ with } \underline{m}_q\in I^q_1 \text{ and } \underline{m}_q=\underline{m}_{q+1},\underline{m}_q\neq\underline{m}_{q-1} \tag{318}$$

$$\tilde{l}^d_{q+1,\underline{m}_q+1}-(\tilde{v}^{d,l}_{q,\underline{m}_q}+\tilde{v}^{d,r}_{q,\underline{m}_q}+\tilde{v}^d_{q,\underline{m}_q})=v^{d,l}_{q,\underline{m}_q}+v^{d,r}_{q,\underline{m}_q}+v^d_{q,\underline{m}_q}$$
$$\forall\, q\in B\backslash\{1,p\} \text{ with } \underline{m}_q\in I^q_1 \text{ and } \underline{m}_q\neq\underline{m}_{q+1},\underline{m}_q\neq\underline{m}_{q-1} \tag{319}$$

$$\tilde{r}^d_{q+1,\overline{m}_q-1}+\tilde{v}^d_{q+1,\overline{m}_q}-(\tilde{v}^{d,l}_{q,\overline{m}_q}+\tilde{v}^{d,r}_{q,\overline{m}_q}+\tilde{v}^d_{q,\overline{m}_q})=v^{d,l}_{q,\overline{m}_q}+v^{d,r}_{q,\overline{m}_q}+v^d_{q,\overline{m}_q}$$
$$\forall\, q\in B\backslash\{1,p\} \text{ with } \overline{m}_q\in I^q_1 \text{ and } \overline{m}_q=\overline{m}_{q+1},\overline{m}_q\neq\overline{m}_{q-1} \tag{320}$$

$$\tilde{r}^d_{q+1,\overline{m}_q-1}-(\tilde{v}^{d,l}_{q,\overline{m}_q}+\tilde{v}^{d,r}_{q,\overline{m}_q}+\tilde{v}^d_{q,\overline{m}_q})=v^{d,l}_{q,\overline{m}_q}+v^{d,r}_{q,\overline{m}_q}+v^d_{q,\overline{m}_q}$$
$$\forall\, q\in B\backslash\{1,p\} \text{ with } \overline{m}_q\in I^q_1 \text{ and } \overline{m}_q\neq\overline{m}_{q+1},\overline{m}_q\neq\overline{m}_{q-1} \tag{321}$$

$$\tilde{l}^d_{q+1,i+1}+\tilde{r}^d_{q+1,i-1}+\tilde{v}^d_{q+1,i}-(\tilde{e}^u_{qi1}+\tilde{t}^d_{qi})=e^u_{qi1}+t^d_{qi} \quad \forall\, q\in B\backslash\{p\}, i\in I^q_2\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } \underline{m}_{q+1}\leq i\leq\overline{m}_{q+1} \tag{322}$$

$$\tilde{l}^d_{q+1,i+1}+\tilde{r}^d_{q+1,i-1}-(\tilde{e}^u_{qi1}+\tilde{t}^d_{qi})=e^u_{qi1}+t^d_{qi} \quad \forall\, q\in B\backslash\{p\}, i\in I^q_2\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } i<\underline{m}_{q+1} \text{ or } i>\overline{m}_{q+1} \tag{323}$$

$$\tilde{l}^d_{q+1,\underline{m}_q+1}+\tilde{v}^d_{q+1,\underline{m}_q}-(\tilde{e}^u_{q,\underline{m}_q,1}+\tilde{t}^d_{q,\underline{m}_q})=e^u_{q,\underline{m}_q,1}+t^d_{q,\underline{m}_q} \quad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^q_2 \text{ and } \underline{m}_q=\underline{m}_{q+1} \tag{324}$$

$$\tilde{l}^d_{q+1,\underline{m}_q+1}-(\tilde{e}^u_{q,\underline{m}_q,1}+\tilde{t}^d_{q,\underline{m}_q})=e^u_{q,\underline{m}_q,1}+t^d_{q,\underline{m}_q} \quad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^q_2 \text{ and } \underline{m}_q\neq\underline{m}_{q+1} \tag{325}$$

$$\tilde{r}^d_{q+1,\overline{m}_q-1}+\tilde{v}^d_{q+1,\overline{m}_q}-(\tilde{e}^u_{q,\overline{m}_q,1}+\tilde{t}^d_{q,\overline{m}_q})=e^u_{q,\overline{m}_q,1}+t^d_{q,\overline{m}_q} \quad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^q_2 \text{ and } \overline{m}_q=\overline{m}_{q+1} \tag{326}$$

$$\tilde{r}^d_{q+1,\overline{m}_q-1}-(\tilde{e}^u_{q,\overline{m}_q,1}+\tilde{t}^d_{q,\overline{m}_q})=e^u_{q,\overline{m}_q,1}+t^d_{q,\overline{m}_q} \quad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^q_2 \text{ and } \overline{m}_q\neq\overline{m}_{q+1} \tag{327}$$

$$\tilde{l}^d_{p+1,i+1}+\tilde{r}^d_{p+1,i-1}-(\tilde{e}^u_{pi1}+\tilde{t}^d_{pi})=e^u_{pi1}+t^d_{pi} \quad \forall\, i\in I^p_2\backslash\{\underline{m}_p,\overline{m}_p\} \tag{328}$$

$$\tilde{l}^d_{p+1,\underline{m}_p+1}-(\tilde{e}^u_{p,\underline{m}_p,1}+\tilde{t}^d_{p,\underline{m}_p})=e^u_{p,\underline{m}_p,1}+t^d_{p,\underline{m}_p} \quad \text{if } \underline{m}_p\in I^p_2 \tag{329}$$

$$\tilde{r}^d_{p+1,\overline{m}_p-1}-(\tilde{e}^u_{p,\overline{m}_p,1}+\tilde{t}^d_{p,\overline{m}_p})=e^u_{p,\overline{m}_p,1}+t^d_{p,\overline{m}_p} \quad \text{if } \overline{m}_p\in I^p_2 \tag{330}$$

$$\tilde{l}^d_{q+1,i+1}+\tilde{r}^d_{q+1,i-1}+\tilde{v}^d_{q+1,i}-(\tilde{e}^u_{qi1}+\tilde{e}^u_{qi2}+\tilde{t}^d_{qi})=e^u_{qi1}+e^u_{qi2}+t^d_{qi}$$
$$\forall\, q\in B\backslash\{p\}, i\in(I^q_3\cup I^q_4)\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } \underline{m}_{q+1}\leq i\leq\overline{m}_{q+1} \tag{331}$$

$$\tilde{l}^d_{q+1,i+1}+\tilde{r}^d_{q+1,i-1}-(\tilde{e}^u_{qi1}+\tilde{e}^u_{qi2}+\tilde{t}^d_{qi})=e^u_{qi1}+e^u_{qi2}+t^d_{qi}$$
$$\forall\, q\in B\backslash\{p\}, i\in(I^q_3\cup I^q_4)\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } i<\underline{m}_{q+1} \text{ or } i>\overline{m}_{q+1} \tag{332}$$

$$\tilde{l}^d_{q+1,\underline{m}_q+1}+\tilde{v}^d_{q+1,\underline{m}_q}-(\tilde{e}^u_{q,\underline{m}_q,1}+\tilde{e}^u_{q,\underline{m}_q,2}+\tilde{t}^d_{q,\underline{m}_q})=e^u_{q,\underline{m}_q,1}+e^u_{q,\underline{m}_q,2}+t^d_{q,\underline{m}_q}$$
$$\forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^q_3\cup I^q_4 \text{ and } \underline{m}_q=\underline{m}_{q+1} \tag{333}$$

$$\tilde{l}^d_{q+1,\underline{m}_q+1} - (\tilde{e}^u_{q,\underline{m}_q,1} + \tilde{e}^u_{q,\underline{m}_q,2} + \tilde{t}^d_{q,\underline{m}_q}) = e^u_{q,\underline{m}_q,1} + e^u_{q,\underline{m}_q,2} + t^d_{q,\underline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_3 \cup I^q_4 \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \quad (334)$$

$$\tilde{r}^d_{q+1,\overline{m}_q-1} + \tilde{v}^d_{q+1,\overline{m}_q} - (\tilde{e}^u_{q,\overline{m}_q,1} + \tilde{e}^u_{q,\overline{m}_q,2} + \tilde{t}^d_{q,\overline{m}_q}) = e^u_{q,\overline{m}_q,1} + e^u_{q,\overline{m}_q,2} + t^d_{q,\overline{m}_q}$$
$$\forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_3 \cup I^q_4 \text{ and } \overline{m}_q = \overline{m}_{q+1} \quad (335)$$

$$\tilde{r}^d_{q+1,\overline{m}_q-1} - (\tilde{e}^u_{q,\overline{m}_q,1} + \tilde{e}^u_{q,\overline{m}_q,2} + \tilde{t}^d_{q,\overline{m}_q}) = e^u_{q,\overline{m}_q,1} + e^u_{q,\overline{m}_q,2} + t^d_{q,\overline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_3 \cup I^q_4 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \quad (336)$$

$$\tilde{l}^d_{p+1,i+1} + \tilde{r}^d_{p+1,i-1} - (\tilde{e}^u_{pi1} + \tilde{e}^u_{pi2} + \tilde{t}^d_{pi}) = e^u_{pi1} + e^u_{pi2} + t^d_{pi} \quad \forall\, i \in (I^p_3 \cup I^p_4)\backslash\{\underline{m}_p, \overline{m}_p\} \quad (337)$$

$$\tilde{l}^d_{p+1,\underline{m}_p+1} - (\tilde{e}^u_{p,\underline{m}_p,1} + \tilde{e}^u_{p,\underline{m}_p,2} + \tilde{t}^d_{p,\underline{m}_p}) = e^u_{p,\underline{m}_p,1} + e^u_{p,\underline{m}_p,2} + t^d_{p,\underline{m}_p} \quad \text{if } \underline{m}_p \in I^p_3 \cup I^p_4 \quad (338)$$

$$\tilde{r}^d_{p+1,\overline{m}_p-1} - (\tilde{e}^u_{p,\overline{m}_p,1} + \tilde{e}^u_{p,\overline{m}_p,2} + \tilde{t}^d_{p,\overline{m}_p}) = e^u_{p,\overline{m}_p,1} + e^u_{p,\overline{m}_p,2} + t^d_{p,\overline{m}_p} \quad \text{if } \overline{m}_p \in I^p_3 \cup I^p_4 \quad (339)$$

$$\tilde{l}^d_{q+1,i+1} + \tilde{r}^d_{q+1,i-1} + \tilde{v}^d_{q+1,i} - (\tilde{e}^u_{qi1} + \tilde{e}^u_{qi3} + \tilde{t}^d_{qi}) = e^u_{qi1} + e^u_{qi3} + t^d_{qi}$$
$$\forall\, q \in B\backslash\{p\}, i \in (I^q_0 \cup I^q_5)\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } \underline{m}_{q+1} \leq i \leq \overline{m}_{q+1} \quad (340)$$

$$\tilde{l}^d_{q+1,i+1} + \tilde{r}^d_{q+1,i-1} - (\tilde{e}^u_{qi1} + \tilde{e}^u_{qi3} + \tilde{t}^d_{qi}) = e^u_{qi1} + e^u_{qi3} + t^d_{qi}$$
$$\forall\, q \in B\backslash\{p\}, i \in (I^q_0 \cup I^q_5)\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \quad (341)$$

$$\tilde{l}^d_{q+1,\underline{m}_q+1} + \tilde{v}^d_{q+1,\underline{m}_q} - (\tilde{e}^u_{q,\underline{m}_q,1} + \tilde{e}^u_{q,\underline{m}_q,3} + \tilde{t}^d_{q,\underline{m}_q}) = e^u_{q,\underline{m}_q,1} + e^u_{q,\underline{m}_q,3} + t^d_{q,\underline{m}_q}$$
$$\forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_0 \cup I^q_5 \text{ and } \underline{m}_q = \underline{m}_{q+1} \quad (342)$$

$$\tilde{l}^d_{q+1,\underline{m}_q+1} - (\tilde{e}^u_{q,\underline{m}_q,1} + \tilde{e}^u_{q,\underline{m}_q,3} + \tilde{t}^d_{q,\underline{m}_q}) = e^u_{q,\underline{m}_q,1} + e^u_{q,\underline{m}_q,3} + t^d_{q,\underline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_0 \cup I^q_5 \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \quad (343)$$

$$\tilde{r}^d_{q+1,\overline{m}_q-1} + \tilde{v}^d_{q+1,\overline{m}_q} - (\tilde{e}^u_{q,\overline{m}_q,1} + \tilde{e}^u_{q,\overline{m}_q,3} + \tilde{t}^d_{q,\overline{m}_q}) = e^u_{q,\overline{m}_q,1} + e^u_{q,\overline{m}_q,3} + t^d_{q,\overline{m}_q}$$
$$\forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_0 \cup I^q_5 \text{ and } \overline{m}_q = \overline{m}_{q+1} \quad (344)$$

$$\tilde{r}^d_{q+1,\overline{m}_q-1} - (\tilde{e}^u_{q,\overline{m}_q,1} + \tilde{e}^u_{q,\overline{m}_q,3} + \tilde{t}^d_{q,\overline{m}_q}) = e^u_{q,\overline{m}_q,1} + e^u_{q,\overline{m}_q,3} + t^d_{q,\overline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_0 \cup I^q_5 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \quad (345)$$

$$\tilde{l}^d_{p+1,i+1} + \tilde{r}^d_{p+1,i-1} - (\tilde{e}^u_{pi1} + \tilde{e}^u_{pi3} + \tilde{t}^d_{pi}) = e^u_{pi1} + e^u_{pi3} + t^d_{pi} \quad \forall\, i \in (I^p_0 \cup I^p_5)\backslash\{\underline{m}_p, \overline{m}_p\} \quad (346)$$

$$\tilde{l}^d_{p+1,\underline{m}_p+1} - (\tilde{e}^u_{p,\underline{m}_p,1} + \tilde{e}^u_{p,\underline{m}_p,3} + \tilde{t}^d_{p,\underline{m}_p}) = e^u_{p,\underline{m}_p,1} + e^u_{p,\underline{m}_p,3} + t^d_{p,\underline{m}_p} \quad \text{if } \underline{m}_p \in I^p_0 \cup I^p_5 \quad (347)$$

$$\tilde{r}^d_{p+1,\overline{m}_p-1} - (\tilde{e}^u_{p,\overline{m}_p,1} + \tilde{e}^u_{p,\overline{m}_p,3} + \tilde{t}^d_{p,\overline{m}_p}) = e^u_{p,\overline{m}_p,1} + e^u_{p,\overline{m}_p,3} + t^d_{p,\overline{m}_p} \quad \text{if } \overline{m}_p \in I^p_0 \cup I^p_5 \quad (348)$$

- Constraints corresponding to vertices $[u,q,i,1]$

$$\tilde{e}^u_{qi1} + \tilde{t}^u_{qi} - (\tilde{v}^u_{qi} + \tilde{v}^{u,l}_{qi} + \tilde{v}^{u,r}_{qi}) = v^u_{qi} + v^{u,l}_{qi} + v^{u,r}_{qi} \quad \forall\, q \in B\backslash\{p\}, i \in I^q_2\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } \underline{m}_{q+1} \leq i \leq \overline{m}_{q+1} \quad (349)$$

$$\tilde{e}^u_{qi1} + \tilde{t}^u_{qi} - (\tilde{v}^{u,l}_{qi} + \tilde{v}^{u,r}_{qi}) = v^{u,l}_{qi} + v^{u,r}_{qi} \quad \forall\, q \in B\backslash\{p\}, i \in I^q_2\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \quad (350)$$

$$\tilde{e}^u_{q,\underline{m}_q,1} + \tilde{t}^u_{q,\underline{m}_q} - (\tilde{v}^u_{q,\underline{m}_q} + \tilde{v}^{u,r}_{q,\underline{m}_q}) = v^u_{q,\underline{m}_q} + v^{u,r}_{q,\underline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_2 \text{ and } \underline{m}_q = \underline{m}_{q+1} \quad (351)$$

$$\tilde{e}^u_{q,\underline{m}_q,1} + \tilde{t}^u_{q,\underline{m}_q} - \tilde{v}^{u,r}_{q,\underline{m}_q} = v^{u,r}_{q,\underline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_2 \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \quad (352)$$

$$\tilde{e}^u_{q,\overline{m}_q,1} + \tilde{t}^u_{q,\overline{m}_q} - (\tilde{v}^u_{q,\overline{m}_q} + \tilde{v}^{u,l}_{q,\overline{m}_q}) = v^u_{q,\overline{m}_q} + v^{u,l}_{q,\overline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_2 \text{ and } \overline{m}_q = \overline{m}_{q+1} \quad (353)$$

$$\tilde{e}^u_{q,\overline{m}_q,1} + \tilde{t}^u_{q,\overline{m}_q} - \tilde{v}^{u,l}_{q,\overline{m}_q} = v^{u,l}_{q,\overline{m}_q} \quad \forall\, q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_2 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \quad (354)$$

$$\tilde{e}^u_{pi1} + \tilde{t}^u_{pi} - (\tilde{v}^{u,l}_{pi} + \tilde{v}^{u,r}_{pi}) = v^{u,l}_{pi} + v^{u,r}_{pi} \quad \forall\, i \in I^p_2\backslash\{\underline{m}_p, \overline{m}_p\} \quad (355)$$

$$\tilde{e}^u_{p,\underline{m}_p,1} + \tilde{t}^u_{p,\underline{m}_p} - \tilde{v}^{u,r}_{p,\underline{m}_p} = v^{u,r}_{p,\underline{m}_p} \quad \text{if } \underline{m}_p \in I^p_2 \quad (356)$$

$$\tilde{e}^u_{p,\overline{m}_p,1} + \tilde{t}^u_{p,\overline{m}_p} - \tilde{v}^{u,l}_{p,\overline{m}_p} = v^{u,l}_{p,\overline{m}_p} \quad \text{if } \overline{m}_p \in I^p_2 \quad (357)$$

$$\tilde{e}^u_{qi1} + \tilde{t}^u_{qi} - \tilde{w}^u_{qi1} = w^u_{qi1} \quad \forall\, q \in B, i \in I^q\backslash(I^q_1 \cup I^q_2) \quad (358)$$

- Constraints corresponding to vertices $[u,q,i,2]$

$$\tilde{e}^u_{qi2}+\tilde{w}^u_{qi1}-(\tilde{v}^u_{qi}+\tilde{v}^{u,l}_{qi}+\tilde{v}^{u,r}_{qi})=v^u_{qi}+v^{u,l}_{qi}+v^{u,r}_{qi} \qquad \forall\, q\in B\backslash\{p\}, i\in I^q_3\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } \underline{m}_{q+1}\leq i\leq \overline{m}_{q+1} \tag{359}$$

$$\tilde{e}^u_{qi2}+\tilde{w}^u_{qi1}-(\tilde{v}^{u,l}_{qi}+\tilde{v}^{u,r}_{qi})=v^{u,l}_{qi}+v^{u,r}_{qi} \qquad \forall\, q\in B\backslash\{p\}, i\in I^q_3\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } i<\underline{m}_{q+1} \text{ or } i>\overline{m}_{q+1} \tag{360}$$

$$\tilde{e}^u_{q,\underline{m}_q,2}+\tilde{w}^u_{q,\underline{m}_q,1}-(\tilde{v}^u_{q,\underline{m}_q}+\tilde{v}^{u,r}_{q,\underline{m}_q})=v^u_{q,\underline{m}_q}+v^{u,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^q_3 \text{ and } \underline{m}_q=\underline{m}_{q+1} \tag{361}$$

$$\tilde{e}^u_{q,\underline{m}_q,2}+\tilde{w}^u_{q,\underline{m}_q,1}-\tilde{v}^{u,r}_{q,\underline{m}_q}=v^{u,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^q_3 \text{ and } \underline{m}_q\neq\underline{m}_{q+1} \tag{362}$$

$$\tilde{e}^u_{q,\overline{m}_q,2}+\tilde{w}^u_{q,\overline{m}_q,1}-(\tilde{v}^u_{q,\overline{m}_q}+\tilde{v}^{u,l}_{q,\overline{m}_q})=v^u_{q,\overline{m}_q}+v^{u,l}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^q_3 \text{ and } \overline{m}_q=\overline{m}_{q+1} \tag{363}$$

$$\tilde{e}^u_{q,\overline{m}_q,2}+\tilde{w}^u_{q,\overline{m}_q,1}-\tilde{v}^{u,l}_{q,\overline{m}_q}=v^{u,l}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^q_3 \text{ and } \overline{m}_q\neq\overline{m}_{q+1} \tag{364}$$

$$\tilde{e}^u_{pi2}+\tilde{w}^u_{pi1}-(\tilde{v}^{u,l}_{pi}+\tilde{v}^{u,r}_{pi})=v^{u,l}_{pi}+v^{u,r}_{pi} \qquad \forall\, i\in I^p_3\backslash\{\underline{m}_p,\overline{m}_p\} \tag{365}$$

$$\tilde{e}^u_{p,\underline{m}_p,2}+\tilde{w}^u_{p,\underline{m}_p,1}-\tilde{v}^{u,r}_{p,\underline{m}_p}=v^{u,r}_{p,\underline{m}_p} \qquad \text{if } \underline{m}_p\in I^p_3 \tag{366}$$

$$\tilde{e}^u_{p,\overline{m}_p,2}+\tilde{w}^u_{p,\overline{m}_p,1}-\tilde{v}^{u,l}_{p,\overline{m}_p}=v^{u,l}_{p,\overline{m}_p} \qquad \text{if } \overline{m}_p\in I^p_3 \tag{367}$$

$$\tilde{e}^u_{qi2}+\tilde{w}^u_{qi1}-\tilde{w}^u_{qi2}=w^u_{qi2} \qquad \forall\, q\in B, i\in I^q_4 \tag{368}$$

$$\tilde{w}^u_{qi1}-\tilde{w}^u_{qi2}=w^u_{qi2} \qquad \forall\, q\in B, i\in I^q_0\cup I^q_5 \tag{369}$$

- Constraints corresponding to vertices $[u,q,i,3]$

$$\tilde{w}^u_{qi2}-(\tilde{v}^u_{qi}+\tilde{v}^{u,l}_{qi}+\tilde{v}^{u,r}_{qi})=v^u_{qi}+v^{u,l}_{qi}+v^{u,r}_{qi} \qquad \forall\, q\in B\backslash\{p\}, i\in I^q_4\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } \underline{m}_{q+1}\leq i\leq \overline{m}_{q+1} \tag{370}$$

$$\tilde{w}^u_{qi2}-(\tilde{v}^{u,l}_{qi}+\tilde{v}^{u,r}_{qi})=v^{u,l}_{qi}+v^{u,r}_{qi} \qquad \forall\, q\in B\backslash\{p\}, i\in I^q_4\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } i<\underline{m}_{q+1} \text{ or } i>\overline{m}_{q+1} \tag{371}$$

$$\tilde{w}^u_{q,\underline{m}_q,2}-(\tilde{v}^u_{q,\underline{m}_q}+\tilde{v}^{u,r}_{q,\underline{m}_q})=v^u_{q,\underline{m}_q}+v^{u,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^q_4 \text{ and } \underline{m}_q=\underline{m}_{q+1} \tag{372}$$

$$\tilde{w}^u_{q,\underline{m}_q,2}-\tilde{v}^{u,r}_{q,\underline{m}_q}=v^{u,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^q_4 \text{ and } \underline{m}_q\neq\underline{m}_{q+1} \tag{373}$$

$$\tilde{w}^u_{q,\overline{m}_q,2}-(\tilde{v}^u_{q,\overline{m}_q}+\tilde{v}^{u,l}_{q,\overline{m}_q})=v^u_{q,\overline{m}_q}+v^{u,l}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^q_4 \text{ and } \overline{m}_q=\overline{m}_{q+1} \tag{374}$$

$$\tilde{w}^u_{q,\overline{m}_q,2}-\tilde{v}^{u,l}_{q,\overline{m}_q}=v^{u,l}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^q_4 \text{ and } \overline{m}_q\neq\overline{m}_{q+1} \tag{375}$$

$$\tilde{w}^u_{pi2}-(\tilde{v}^{u,l}_{pi}+\tilde{v}^{u,r}_{pi})=v^{u,l}_{pi}+v^{u,r}_{pi} \qquad \forall\, i\in I^p_4\backslash\{\underline{m}_p,\overline{m}_p\} \tag{376}$$

$$\tilde{w}^u_{p,\underline{m}_p,2}-\tilde{v}^{u,r}_{p,\underline{m}_p}=v^{u,r}_{p,\underline{m}_p} \qquad \text{if } \underline{m}_p\in I^p_4 \tag{377}$$

$$\tilde{w}^u_{p,\overline{m}_p,2}-\tilde{v}^{u,l}_{p,\overline{m}_p}=v^{u,l}_{p,\overline{m}_p} \qquad \text{if } \overline{m}_p\in I^p_4 \tag{378}$$

$$\tilde{e}^u_{qi3}+\tilde{w}^u_{qi2}-(\tilde{v}^u_{qi}+\tilde{v}^{u,l}_{qi}+\tilde{v}^{u,r}_{qi})=v^u_{qi}+v^{u,l}_{qi}+v^{u,r}_{qi} \qquad \forall\, q\in B\backslash\{p\}, i\in I^q_5\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } \underline{m}_{q+1}\leq i\leq \overline{m}_{q+1} \tag{379}$$

$$\tilde{e}^u_{qi3}+\tilde{w}^u_{qi2}-(\tilde{v}^{u,l}_{qi}+\tilde{v}^{u,r}_{qi})=v^{u,l}_{qi}+v^{u,r}_{qi} \qquad \forall\, q\in B\backslash\{p\}, i\in I^q_5\backslash\{\underline{m}_q,\overline{m}_q\} \text{ with } i<\underline{m}_{q+1} \text{ or } i>\overline{m}_{q+1} \tag{380}$$

$$\tilde{e}^u_{q,\underline{m}_q,3}+\tilde{w}^u_{q,\underline{m}_q,2}-(\tilde{v}^u_{q,\underline{m}_q}+\tilde{v}^{u,r}_{q,\underline{m}_q})=v^u_{q,\underline{m}_q}+v^{u,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^q_5 \text{ and } \underline{m}_q=\underline{m}_{q+1} \tag{381}$$

$$\tilde{e}^u_{q,\underline{m}_q,3}+\tilde{w}^u_{q,\underline{m}_q,2}-\tilde{v}^{u,r}_{q,\underline{m}_q}=v^{u,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \underline{m}_q\in I^q_5 \text{ and } \underline{m}_q\neq\underline{m}_{q+1} \tag{382}$$

$$\tilde{e}^u_{q,\overline{m}_q,3}+\tilde{w}^u_{q,\overline{m}_q,2}-(\tilde{v}^u_{q,\overline{m}_q}+\tilde{v}^{u,l}_{q,\overline{m}_q})=v^u_{q,\overline{m}_q}+v^{u,l}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^q_5 \text{ and } \overline{m}_q=\overline{m}_{q+1} \tag{383}$$

$$\tilde{e}^u_{q,\overline{m}_q,3}+\tilde{w}^u_{q,\overline{m}_q,2}-\tilde{v}^{u,l}_{q,\overline{m}_q}=v^{u,l}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{p\} \text{ with } \overline{m}_q\in I^q_5 \text{ and } \overline{m}_q\neq\overline{m}_{q+1} \tag{384}$$

$$\tilde{e}^u_{pi3}+\tilde{w}^u_{pi2}-(\tilde{v}^{u,l}_{pi}+\tilde{v}^{u,r}_{pi})=v^{u,l}_{pi}+v^{u,r}_{pi} \qquad \forall\, i\in I^p_5\backslash\{\underline{m}_p,\overline{m}_p\} \tag{385}$$

$$\tilde{e}^u_{p,\underline{m}_p,3}+\tilde{w}^u_{p,\underline{m}_p,2}-\tilde{v}^{u,r}_{p,\underline{m}_p}=v^{u,r}_{p,\underline{m}_p} \qquad \text{if } \underline{m}_p\in I^p_5 \tag{386}$$

$$\tilde{e}^u_{p,\overline{m}_p,3}+\tilde{w}^u_{p,\overline{m}_p,2}-\tilde{v}^{u,l}_{p,\overline{m}_p}=v^{u,l}_{p,\overline{m}_p} \qquad \text{if } \overline{m}_p\in I^p_5 \tag{387}$$

$$\tilde{e}^u_{qi3}+\tilde{w}^u_{qi2}-\tilde{w}^u_{qi3}=w^u_{qi3} \qquad \forall\, q\in B, i\in I^q_0 \tag{388}$$

- Constraints corresponding to vertices $[u,q,i,4]$

$$\tilde{w}^u_{qi3} - (\tilde{v}^u_{qi} + \tilde{v}^{u,l}_{qi} + \tilde{v}^{u,r}_{qi}) = v^u_{qi} + v^{u,l}_{qi} + v^{u,r}_{qi} \qquad \forall\ q \in B\backslash\{p\}, i \in I^q_0\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } \underline{m}_{q+1} \le i \le \overline{m}_{q+1} \quad (389)$$

$$\tilde{w}^u_{qi3} - (\tilde{v}^{u,l}_{qi} + \tilde{v}^{u,r}_{qi}) = v^{u,l}_{qi} + v^{u,r}_{qi} \qquad \forall\ q \in B\backslash\{p\}, i \in I^q_0\backslash\{\underline{m}_q, \overline{m}_q\} \text{ with } i < \underline{m}_{q+1} \text{ or } i > \overline{m}_{q+1} \quad (390)$$

$$\tilde{w}^u_{q,\underline{m}_q,3} - (\tilde{v}^u_{q,\underline{m}_q} + \tilde{v}^{u,r}_{q,\underline{m}_q}) = v^u_{q,\underline{m}_q} + v^{u,r}_{q,\underline{m}_q} \qquad \forall\ q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_0 \text{ and } \underline{m}_q = \underline{m}_{q+1} \quad (391)$$

$$\tilde{w}^u_{q,\underline{m}_q,3} - \tilde{v}^{u,r}_{q,\underline{m}_q} = v^{u,r}_{q,\underline{m}_q} \qquad \forall\ q \in B\backslash\{p\} \text{ with } \underline{m}_q \in I^q_0 \text{ and } \underline{m}_q \neq \underline{m}_{q+1} \quad (392)$$

$$\tilde{w}^u_{q,\overline{m}_q,3} - (\tilde{v}^u_{q,\overline{m}_q} + \tilde{v}^{u,l}_{q,\overline{m}_q}) = v^u_{q,\overline{m}_q} + v^{u,l}_{q,\overline{m}_q} \qquad \forall\ q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_0 \text{ and } \overline{m}_q = \overline{m}_{q+1} \quad (393)$$

$$\tilde{w}^u_{q,\overline{m}_q,3} - \tilde{v}^{u,l}_{q,\overline{m}_q} = v^{u,l}_{q,\overline{m}_q} \qquad \forall\ q \in B\backslash\{p\} \text{ with } \overline{m}_q \in I^q_0 \text{ and } \overline{m}_q \neq \overline{m}_{q+1} \quad (394)$$

$$\tilde{w}^u_{pi3} - (\tilde{v}^{u,l}_{pi} + \tilde{v}^{u,r}_{pi}) = v^{u,l}_{pi} + v^{u,r}_{pi} \qquad \forall\ i \in I^p_0\backslash\{\underline{m}_p, \overline{m}_p\} \quad (395)$$

$$\tilde{w}^u_{p,\underline{m}_p,3} - \tilde{v}^{u,r}_{p,\underline{m}_p} = v^{u,r}_{p,\underline{m}_p} \qquad \text{if } \underline{m}_p \in I^p_0 \quad (396)$$

$$\tilde{w}^u_{p,\overline{m}_p,3} - \tilde{v}^{u,l}_{p,\overline{m}_p} = v^{u,l}_{p,\overline{m}_p} \qquad \text{if } \overline{m}_p \in I^p_0 \quad (397)$$

- Constraints corresponding to vertices $[d,q,i,1]$

$$\tilde{e}^d_{1i1} + \tilde{t}^d_{1i} - (\tilde{v}^{d,l}_{1i} + \tilde{v}^{d,r}_{1i}) = v^{d,l}_{1i} + v^{d,r}_{1i} \qquad \forall\ i \in I^1_2\backslash\{\overline{m}_1\} \quad (398)$$

$$\tilde{e}^d_{1,\overline{m}_1,1} + \tilde{t}^d_{1,\overline{m}_1} - \tilde{v}^{d,l}_{1,\overline{m}_1} = v^{d,l}_{1,\overline{m}_1} \qquad \text{if } \overline{m}_1 \in I^1_2 \quad (399)$$

$$\tilde{e}^d_{qi1} + \tilde{t}^d_{qi} - (\tilde{v}^d_{qi} + \tilde{v}^{d,l}_{qi} + \tilde{v}^{d,r}_{qi}) = v^d_{qi} + v^{d,l}_{qi} + v^{d,r}_{qi} \qquad \forall\ q \in B\backslash\{1\}, i \in I^q_2\backslash\{\underline{m}_q, \overline{m}_q\} \quad (400)$$

$$\tilde{e}^d_{q,\underline{m}_q,1} + \tilde{t}^d_{q,\underline{m}_q} - (\tilde{v}^d_{q,\underline{m}_q} + \tilde{v}^{d,l}_{q,\underline{m}_q} + \tilde{v}^{d,r}_{q,\underline{m}_q}) = v^d_{q,\underline{m}_q} + v^{d,l}_{q,\underline{m}_q} + v^{d,r}_{q,\underline{m}_q} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_2 \text{ and } \underline{m}_{q-1} < \underline{m}_q \quad (401)$$

$$\tilde{e}^d_{q,\underline{m}_q,1} + \tilde{t}^d_{q,\underline{m}_q} - (\tilde{v}^d_{q,\underline{m}_q} + \tilde{v}^{d,r}_{q,\underline{m}_q}) = v^d_{q,\underline{m}_q} + v^{d,r}_{q,\underline{m}_q} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_2 \text{ and } \underline{m}_{q-1} = \underline{m}_q \quad (402)$$

$$\tilde{e}^d_{q,\overline{m}_q,1} + \tilde{t}^d_{q,\overline{m}_q} - (\tilde{v}^d_{q,\overline{m}_q} + \tilde{v}^{d,l}_{q,\overline{m}_q} + \tilde{v}^{d,r}_{q,\overline{m}_q}) = v^d_{q,\overline{m}_q} + v^{d,l}_{q,\overline{m}_q} + v^{d,r}_{q,\overline{m}_q} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_2 \text{ and } \overline{m}_{q-1} > \overline{m}_q \quad (403)$$

$$\tilde{e}^d_{q,\overline{m}_q,1} + \tilde{t}^d_{q,\overline{m}_q} - (\tilde{v}^d_{q,\overline{m}_q} + \tilde{v}^{d,l}_{q,\overline{m}_q}) = v^d_{q,\overline{m}_q} + v^{d,l}_{q,\overline{m}_q} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_2 \text{ and } \overline{m}_{q-1} = \overline{m}_q \quad (404)$$

$$\tilde{e}^d_{qi1} + \tilde{t}^d_{qi} - \tilde{w}^d_{qi1} = w^d_{qi1} \qquad \forall\ q \in B, i \in I^q\backslash(I^q_1 \cup I^q_2) \quad (405)$$

- Constraints corresponding to vertices $[d,q,i,2]$

$$\tilde{e}^d_{1i2} + \tilde{w}^d_{1i1} - (\tilde{v}^{d,l}_{1i} + \tilde{v}^{d,r}_{1i}) = v^{d,l}_{1i} + v^{d,r}_{1i} \qquad \forall\ i \in I^1_3\backslash\{\overline{m}_1\} \quad (406)$$

$$\tilde{e}^d_{1,\overline{m}_1,2} + \tilde{w}^d_{1,\overline{m}_1,1} - \tilde{v}^{d,l}_{1,\overline{m}_1} = v^{d,l}_{1,\overline{m}_1} \qquad \text{if } \overline{m}_1 \in I^1_3 \quad (407)$$

$$\tilde{e}^d_{qi2} + \tilde{w}^d_{qi1} - (\tilde{v}^d_{qi} + \tilde{v}^{d,l}_{qi} + \tilde{v}^{d,r}_{qi}) = v^d_{qi} + v^{d,l}_{qi} + v^{d,r}_{qi} \qquad \forall\ q \in B\backslash\{1\}, i \in I^q_3\backslash\{\underline{m}_q, \overline{m}_q\} \quad (408)$$

$$\tilde{e}^d_{q,\underline{m}_q,2} + \tilde{w}^d_{q,\underline{m}_q,1} - (\tilde{v}^d_{q,\underline{m}_q} + \tilde{v}^{d,l}_{q,\underline{m}_q} + \tilde{v}^{d,r}_{q,\underline{m}_q}) = v^d_{q,\underline{m}_q} + v^{d,l}_{q,\underline{m}_q} + v^{d,r}_{q,\underline{m}_q} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_3 \text{ and } \underline{m}_{q-1} < \underline{m}_q \quad (409)$$

$$\tilde{e}^d_{q,\underline{m}_q,2} + \tilde{w}^d_{q,\underline{m}_q,1} - \tilde{v}^d_{q,\underline{m}_q} + \tilde{v}^{d,r}_{q,\underline{m}_q} = v^d_{q,\underline{m}_q} + v^{d,r}_{q,\underline{m}_q} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \underline{m}_q \in I^q_3 \text{ and } \underline{m}_{q-1} = \underline{m}_q \quad (410)$$

$$\tilde{e}^d_{q,\overline{m}_q,2} + \tilde{w}^d_{q,\overline{m}_q,1} - (\tilde{v}^d_{q,\overline{m}_q} + \tilde{v}^{d,l}_{q,\overline{m}_q} + \tilde{v}^{d,r}_{q,\overline{m}_q}) = v^d_{q,\overline{m}_q} + v^{d,l}_{q,\overline{m}_q} + v^{d,r}_{q,\overline{m}_q} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_3 \text{ and } \overline{m}_{q-1} > \overline{m}_q \quad (411)$$

$$\tilde{e}^d_{q,\overline{m}_q,2} + \tilde{w}^d_{q,\overline{m}_q,1} - (\tilde{v}^d_{q,\overline{m}_q} + \tilde{v}^{d,l}_{q,\overline{m}_q}) = v^d_{q,\overline{m}_q} + v^{d,l}_{q,\overline{m}_q} \qquad \forall\ q \in B\backslash\{1\} \text{ with } \overline{m}_q \in I^q_3 \text{ and } \overline{m}_{q-1} = \overline{m}_q \quad (412)$$

$$\tilde{w}^d_{qi1} - \tilde{w}^d_{qi2} = w^d_{qi2} \qquad \forall\ q \in B, i \in I^q_0 \cup I^q_4 \quad (413)$$

$$\tilde{e}^d_{qi2} + \tilde{w}^d_{qi1} - \tilde{w}^d_{qi2} = w^d_{qi2} \qquad \forall\ q \in B, i \in I^q_5 \quad (414)$$

- Constraints corresponding to vertices $[d,q,i,3]$

$$\tilde{e}^d_{1i3}+\tilde{w}^d_{1i2}-(\tilde{v}^{d,l}_{1i}+\tilde{v}^{d,r}_{1i})=v^{d,l}_{1i}+v^{d,r}_{1i} \qquad \forall\, i\in I^1_4\backslash\{\overline{m}_1\} \quad (415)$$

$$\tilde{e}^d_{1,\overline{m}_1,3}+\tilde{w}^d_{1,\overline{m}_1,2}-\tilde{v}^{d,l}_{1,\overline{m}_1}=v^{d,l}_{1,\overline{m}_1} \qquad \text{if } \overline{m}_1\in I^1_4 \quad (416)$$

$$\tilde{e}^d_{qi3}+\tilde{w}^d_{qi2}-(\tilde{v}^d_{qi}+\tilde{v}^{d,l}_{qi}+\tilde{v}^{d,r}_{qi})=v^d_{qi}+v^{d,l}_{qi}+v^{d,r}_{qi} \qquad \forall\, q\in B\backslash\{1\}, i\in I^q_4\backslash\{\underline{m}_q,\overline{m}_q\} \quad (417)$$

$$\tilde{e}^d_{q,\underline{m}_q,3}+\tilde{w}^d_{q,\underline{m}_q,2}-(\tilde{v}^d_{q,\underline{m}_q}+\tilde{v}^{d,l}_{q,\underline{m}_q}+\tilde{v}^{d,r}_{q,\underline{m}_q})=v^d_{q,\underline{m}_q}+v^{d,l}_{q,\underline{m}_q}+v^{d,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \underline{m}_q\in I^q_4 \text{ and } \underline{m}_{q-1}<\underline{m}_q \quad (418)$$

$$\tilde{e}^d_{q,\underline{m}_q,3}+\tilde{w}^d_{q,\underline{m}_q,2}-\tilde{v}^d_{q,\underline{m}_q}+\tilde{v}^{d,r}_{q,\underline{m}_q}=v^d_{q,\underline{m}_q}+v^{d,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \underline{m}_q\in I^q_4 \text{ and } \underline{m}_{q-1}=\underline{m}_q \quad (419)$$

$$\tilde{e}^d_{q,\overline{m}_q,3}+\tilde{w}^d_{q,\overline{m}_q,2}-(\tilde{v}^d_{q,\overline{m}_q}+\tilde{v}^{d,l}_{q,\overline{m}_q}+\tilde{v}^{d,r}_{q,\overline{m}_q})=v^d_{q,\overline{m}_q}+v^{d,l}_{q,\overline{m}_q}+v^{d,r}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \overline{m}_q\in I^q_4 \text{ and } \overline{m}_{q-1}>\overline{m}_q \quad (420)$$

$$\tilde{e}^d_{q,\overline{m}_q,3}+\tilde{w}^d_{q,\overline{m}_q,2}-(\tilde{v}^d_{q,\overline{m}_q}+\tilde{v}^{d,l}_{q,\overline{m}_q})=v^d_{q,\overline{m}_q}+v^{d,l}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \overline{m}_q\in I^q_4 \text{ and } \overline{m}_{q-1}=\overline{m}_q \quad (421)$$

$$\tilde{w}^d_{1i2}-(\tilde{v}^{d,l}_{1i}+\tilde{v}^{d,r}_{1i})=v^{d,l}_{1i}+v^{d,r}_{1i} \qquad \forall\, i\in I^1_5\backslash\{\overline{m}_1\} \quad (422)$$

$$\tilde{w}^d_{1,\overline{m}_1,2}-\tilde{v}^{d,l}_{1,\overline{m}_1}=v^{d,l}_{1,\overline{m}_1} \qquad \text{if } \overline{m}_1\in I^1_5 \quad (423)$$

$$\tilde{w}^d_{qi2}-(\tilde{v}^d_{qi}+\tilde{v}^{d,l}_{qi}+\tilde{v}^{d,r}_{qi})=v^d_{qi}+v^{d,l}_{qi}+v^{d,r}_{qi} \qquad \forall\, q\in B\backslash\{1\}, i\in I^q_5\backslash\{\underline{m}_q,\overline{m}_q\} \quad (424)$$

$$\tilde{w}^d_{q,\underline{m}_q,2}-(\tilde{v}^d_{q,\underline{m}_q}+\tilde{v}^{d,l}_{q,\underline{m}_q}+\tilde{v}^{d,r}_{q,\underline{m}_q})=v^d_{q,\underline{m}_q}+v^{d,l}_{q,\underline{m}_q}+v^{d,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \underline{m}_q\in I^q_5 \text{ and } \underline{m}_{q-1}<\underline{m}_q \quad (425)$$

$$\tilde{w}^d_{q,\underline{m}_q,2}-(\tilde{v}^d_{q,\underline{m}_q}+\tilde{v}^{d,r}_{q,\underline{m}_q})=v^d_{q,\underline{m}_q}+v^{d,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \underline{m}_q\in I^q_5 \text{ and } \underline{m}_{q-1}=\underline{m}_q \quad (426)$$

$$\tilde{w}^d_{q,\overline{m}_q,2}-(\tilde{v}^d_{q,\overline{m}_q}+\tilde{v}^{d,l}_{q,\overline{m}_q}+\tilde{v}^{d,r}_{q,\overline{m}_q})=v^d_{q,\overline{m}_q}+v^{d,l}_{q,\overline{m}_q}+v^{d,r}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \overline{m}_q\in I^q_5 \text{ and } \overline{m}_{q-1}>\overline{m}_q \quad (427)$$

$$\tilde{w}^d_{q,\overline{m}_q,2}-(\tilde{v}^d_{q,\overline{m}_q}+\tilde{v}^{d,l}_{q,\overline{m}_q})=v^d_{q,\overline{m}_q}+v^{d,l}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \overline{m}_q\in I^q_3 \text{ and } \overline{m}_{q-1}=\overline{m}_q \quad (428)$$

$$\tilde{e}^d_{qi3}+\tilde{w}^d_{qi2}-\tilde{w}^d_{qi3}=w^d_{qi3} \qquad \forall\, q\in B, i\in I^q_0 \quad (429)$$

- Constraints corresponding to vertices $[d,q,i,4]$

$$\tilde{w}^d_{1i3}-(\tilde{v}^{d,l}_{1i}+\tilde{v}^{d,r}_{1i})=v^{d,l}_{1i}+v^{d,r}_{1i} \qquad \forall\, i\in I^1_0\backslash\{\overline{m}_1\} \quad (430)$$

$$\tilde{w}^d_{1,\overline{m}_1,3}-\tilde{v}^{d,l}_{1,\overline{m}_1}=v^{d,l}_{1,\overline{m}_1} \qquad \text{if } \overline{m}_1\in I^1_0 \quad (431)$$

$$\tilde{w}^d_{qi3}-(\tilde{v}^d_{qi}+\tilde{v}^{d,l}_{qi}+\tilde{v}^{d,r}_{qi})=v^d_{qi}+v^{d,l}_{qi}+v^{d,r}_{qi} \qquad \forall\, q\in B\backslash\{1\}, i\in I^q_0\backslash\{\underline{m}_q,\overline{m}_q\} \quad (432)$$

$$\tilde{w}^d_{q,\underline{m}_q,3}-(\tilde{v}^d_{q,\underline{m}_q}+\tilde{v}^{d,l}_{q,\underline{m}_q}+\tilde{v}^{d,r}_{q,\underline{m}_q})=v^d_{q,\underline{m}_q}+v^{d,l}_{q,\underline{m}_q}+v^{d,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \underline{m}_q\in I^q_0 \text{ and } \underline{m}_{q-1}<\underline{m}_q \quad (433)$$

$$\tilde{w}^d_{q,\underline{m}_q,3}-(\tilde{v}^d_{q,\underline{m}_q}+\tilde{v}^{d,r}_{q,\underline{m}_q})=v^d_{q,\underline{m}_q}+v^{d,r}_{q,\underline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \underline{m}_q\in I^q_0 \text{ and } \underline{m}_{q-1}=\underline{m}_q \quad (434)$$

$$\tilde{w}^d_{q,\overline{m}_q,3}-(\tilde{v}^d_{q,\overline{m}_q}+\tilde{v}^{d,l}_{q,\overline{m}_q}+\tilde{v}^{d,r}_{q,\overline{m}_q})=v^d_{q,\overline{m}_q}+v^{d,l}_{q,\overline{m}_q}+v^{d,r}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \overline{m}_q\in I^q_0 \text{ and } \overline{m}_{q-1}>\overline{m}_q \quad (435)$$

$$\tilde{w}^d_{q,\overline{m}_q,3}-(\tilde{v}^d_{q,\overline{m}_q}+\tilde{v}^{d,l}_{q,\overline{m}_q})=v^d_{q,\overline{m}_q}+v^{d,l}_{q,\overline{m}_q} \qquad \forall\, q\in B\backslash\{1\} \text{ with } \overline{m}_q\in I^q_0 \text{ and } \overline{m}_{q-1}=\overline{m}_q \quad (436)$$

- Constraints to link variables

$$\tilde{r}^r_{qi}\le M\cdot r^r_{qi} \qquad \forall\,(q,i)\in F\times(I^q\backslash\{\overline{m}^q-1,\overline{m}^q\}) \quad (437)$$

$$\tilde{r}^d_{qi}\le M\cdot r^d_{qi} \qquad \forall\,(q,i)\in(F\backslash\{1\})\times(I^q\backslash\{\overline{m}^q\}) \quad (438)$$

$$\tilde{r}^u_{qi}\le M\cdot r^u_{qi} \qquad \forall\,(q,i)\in(F\backslash\{p+1\})\times(I^q\backslash\{\overline{m}^q\}) \quad (439)$$

$$\tilde{l}^l_{qi}\le M\cdot l^l_{qi} \qquad \forall\,(q,i)\in F\times((I^q\backslash\{\underline{m}^q,\underline{m}^q+1\})\cup\{(1,2)\}) \quad (440)$$

$$\tilde{l}^d_{qi}\le M\cdot l^d_{qi} \qquad \forall\,(q,i)\in(F\backslash\{1\})\times(I^q\backslash\{\underline{m}^q\}) \quad (441)$$

$$\tilde{l}^u_{qi}\le M\cdot l^u_{qi} \qquad \forall\,(q,i)\in(F\backslash\{p+1\})\times(I^q\backslash\{\underline{m}^q\}) \quad (442)$$

$$\tilde{e}^u_{qi1} \leq M \cdot e^u_{qi1} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{443}$$

$$\tilde{e}^d_{qi1} \leq M \cdot e^d_{qi1} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{444}$$

$$\tilde{e}^u_{qi2} \leq M \cdot e^u_{qi2} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_3 \cup I^q_4) \tag{445}$$

$$\tilde{e}^d_{qi2} \leq M \cdot e^d_{qi2} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_3 \cup I^q_5) \tag{446}$$

$$\tilde{e}^u_{qi3} \leq M \cdot e^u_{qi3} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_0 \cup I^q_5) \tag{447}$$

$$\tilde{e}^d_{qi3} \leq M \cdot e^d_{qi3} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_0 \cup I^q_4) \tag{448}$$

$$\tilde{t}^u_{qi} \leq M \cdot t^u_{qi} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{449}$$

$$\tilde{t}^d_{qi} \leq M \cdot t^d_{qi} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{450}$$

$$\tilde{w}^u_{qis} \leq M \cdot w^u_{qis} \qquad \forall\,(q,i,s) \in B \times ((I^q_3 \times \{1\}) \cup ((I^q_4 \cup I^q_5) \times \{1,2\}) \cup (I^q_0 \times \{1,2,3\})) \tag{451}$$

$$\tilde{w}^d_{qis} \leq M \cdot w^d_{qis} \qquad \forall\,(q,i,s) \in B \times ((I^q_3 \times \{1\}) \cup ((I^q_4 \cup I^q_5) \times \{1,2\}) \cup (I^q_0 \times \{1,2,3\})) \tag{452}$$

$$\tilde{v}^u_{qi} \leq M \cdot v^u_{qi} \qquad \forall\,(q,i) \in (B \backslash \{p\}) \times I^q \tag{453}$$

$$\tilde{v}^{u,l}_{qi} \leq M \cdot v^{u,l}_{qi} \qquad \forall\,(q,i) \in B \times (I^q \backslash \{\underline{m}_q\}) \tag{454}$$

$$\tilde{v}^{u,r}_{qi} \leq M \cdot v^{u,r}_{qi} \qquad \forall\,(q,i) \in B \times (I^q \backslash \{\overline{m}_q\}) \tag{455}$$

$$\tilde{v}^d_{qi} \leq M \cdot v^d_{qi} \qquad \forall\,(q,i) \in (B \backslash \{1\}) \times I^q \tag{456}$$

$$\tilde{v}^{d,l}_{qi} \leq M \cdot v^{d,l}_{qi} \qquad \forall\,(q,i) \in B \times ((I^q \backslash \{\underline{m}_q\}) \cup \{(1,1)\}) \tag{457}$$

$$\tilde{v}^{d,r}_{qi} \leq M \cdot v^{d,r}_{qi} \qquad \forall\,(q,i) \in B \times (I^q \backslash \{\overline{m}_q\}) \tag{458}$$

$$\tilde{y}^0_\alpha \leq M \cdot y^0_\alpha \qquad \forall\,\alpha \in \{l,r,u\} \tag{459}$$

$$\tilde{y}^l_0 \leq M \cdot y^l_0 \tag{460}$$

Constraints for the Domains of the Variables:

$$r^r_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in F \times (I^q \backslash \{\overline{m}^q - 1, \overline{m}^q\}) \tag{461}$$

$$r^d_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{1\}) \times (I^q \backslash \{\overline{m}^q\}) \tag{462}$$

$$r^u_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash \{\overline{m}^q\}) \tag{463}$$

$$l^l_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in F \times ((I^q \backslash \{\underline{m}^q, \underline{m}^q + 1\}) \cup \{(1,2)\}) \tag{464}$$

$$l^d_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{1\}) \times (I^q \backslash \{\underline{m}^q\}) \tag{465}$$

$$l^u_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash \{\underline{m}^q\}) \tag{466}$$

$$e^u_{qi1} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{467}$$

$$e^d_{qi1} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{468}$$

$$e^u_{qi2} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_3 \cup I^q_4) \tag{469}$$

$$e^d_{qi2} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_3 \cup I^q_5) \tag{470}$$

$$e^u_{qi3} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_0 \cup I^q_5) \tag{471}$$

$$e^d_{qi3} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_0 \cup I^q_4) \tag{472}$$

$$t^u_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{473}$$

$$t^d_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{474}$$

$$w^u_{qis} \in \{0,1\} \qquad \forall\,(q,i,s) \in B \times ((I^q_3 \times \{1\}) \cup ((I^q_4 \cup I^q_5) \times \{1,2\}) \cup (I^q_0 \times \{1,2,3\})) \tag{475}$$

$$w^d_{qis} \in \{0,1\} \qquad \forall\,(q,i,s) \in B \times ((I^q_3 \times \{1\}) \cup ((I^q_4 \cup I^q_5) \times \{1,2\}) \cup (I^q_0 \times \{1,2,3\})) \tag{476}$$

$$v^u_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in (B \backslash \{p\}) \times I^q \tag{477}$$

$$v^{u,l}_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in B \times (I^q \backslash \{\underline{m}_q\}) \tag{478}$$

$$v^{u,r}_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in B \times (I^q \backslash \{\overline{m}_q\}) \tag{479}$$

$$v^d_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in (B \backslash \{1\}) \times I^q \tag{480}$$

$$v^{d,l}_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in B \times ((I^q \backslash \{\underline{m}_q\}) \cup \{(1,1)\}) \tag{481}$$

$$v^{d,r}_{qi} \in \{0,1\} \qquad \forall\,(q,i) \in B \times (I^q \backslash \{\overline{m}_q\}) \tag{482}$$

$$y^0_\alpha \in \{0,1\} \qquad \forall\,\alpha \in \{l,r,u\} \tag{483}$$

$$y^l_0 \in \{0,1\} \tag{484}$$

$$\tilde{r}^r_{qi} \geq 0 \qquad \forall\,(q,i) \in F \times (I^q \backslash \{\overline{m}^q - 1, \overline{m}^q\}) \tag{485}$$

$$\tilde{r}^d_{qi} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{1\}) \times (I^q \backslash \{\overline{m}^q\}) \tag{486}$$

$$\tilde{r}^u_{qi} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash \{\overline{m}^q\}) \tag{487}$$

$$\tilde{l}^l_{qi} \geq 0 \qquad \forall\,(q,i) \in F \times ((I^q \backslash \{\underline{m}^q, \underline{m}^q + 1\}) \cup \{(1,2)\}) \tag{488}$$

$$\tilde{l}^d_{qi} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{1\}) \times (I^q \backslash \{\underline{m}^q\}) \tag{489}$$

$$\tilde{l}^u_{qi} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash \{\underline{m}^q\}) \tag{490}$$

$$\tilde{e}^u_{qi1} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{491}$$

$$\tilde{e}^d_{qi1} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{492}$$

$$\tilde{e}^u_{qi2} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_3 \cup I^q_4) \tag{493}$$

$$\tilde{e}^d_{qi2} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_3 \cup I^q_5) \tag{494}$$

$$\tilde{e}^u_{qi3} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_0 \cup I^q_5) \tag{495}$$

$$\tilde{e}^d_{qi3} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q_0 \cup I^q_4) \tag{496}$$

$$\tilde{t}^u_{qi} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{497}$$

$$\tilde{t}^d_{qi} \geq 0 \qquad \forall\,(q,i) \in (F \backslash \{p+1\}) \times (I^q \backslash I^q_1) \tag{498}$$

$$\tilde{w}^u_{qis} \geq 0 \qquad \forall\,(q,i,s) \in B \times ((I^q_3 \times \{1\}) \cup ((I^q_4 \cup I^q_5) \times \{1,2\}) \cup (I^q_0 \times \{1,2,3\})) \tag{499}$$

$$\tilde{w}^d_{qis} \geq 0 \qquad \forall\,(q,i,s) \in B \times ((I^q_3 \times \{1\}) \cup ((I^q_4 \cup I^q_5) \times \{1,2\}) \cup (I^q_0 \times \{1,2,3\})) \tag{500}$$

$$\tilde{v}^u_{qi} \geq 0 \qquad \forall\,(q,i) \in (B \backslash \{p\}) \times I^q \tag{501}$$

$$\tilde{v}^{u,l}_{qi} \geq 0 \qquad \forall\,(q,i) \in B \times (I^q \backslash \{\underline{m}_q\}) \tag{502}$$

$$\tilde{v}^{u,r}_{qi} \geq 0 \qquad \forall\,(q,i) \in B \times (I^q \backslash \{\overline{m}_q\}) \tag{503}$$

$$\tilde{v}^d_{qi} \geq 0 \qquad \forall\,(q,i) \in (B \backslash \{1\}) \times I^q \tag{504}$$

$$\tilde{v}^{d,l}_{qi} \geq 0 \qquad \forall\,(q,i) \in B \times ((I^q \backslash \{\underline{m}_q\}) \cup \{(1,1)\}) \tag{505}$$

$$\tilde{v}^{d,r}_{qi} \geq 0 \qquad \forall\,(q,i) \in B \times (I^q \backslash \{\overline{m}_q\}) \tag{506}$$

$$\tilde{y}^0_\alpha \geq 0 \qquad \forall\,\alpha \in \{l,r,u\} \tag{507}$$

$$\tilde{y}^l_0 \geq 0 \tag{508}$$

Part V:

Picker Routing in Narrow-Aisle Warehouses

WORKING PAPER SERIES

Order Picking in Narrow-Aisle Warehouses: A Fast Approach to Minimize Waiting Times

Sandra Hahn/André Scholz

Working Paper No. 6/2017

OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG

FACULTY OF ECONOMICS
AND MANAGEMENT

Impressum (§ 5 TMG)
Herausgeber:
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Der Dekan

Verantwortlich für diese Ausgabe:
Sandra Hahn, André Scholz
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Postfach 4120
39016 Magdeburg
Germany

http://www.fww.ovgu.de/femm

Bezug über den Herausgeber
ISSN 1615-4274

Order Picking in Narrow-Aisle Warehouses: A Fast Approach to Minimize Waiting Times

S. Hahn, A. Scholz

Abstract

Mail order companies like Zalando or Amazon reported a significant increase regarding the number of incoming customer orders in recent years. Customers are served from a central distribution center (warehouse) where requested items of the orders have to be retrieved (picked) from their storage locations. The picking process is performed by human operators (order pickers) who are employed on a large scale in order to enable a fast processing of the orders. However, due to limited space, aisles are often very narrow in warehouses, and order pickers cannot pass or overtake each other. Thus, an order picker may have to wait until another picker has performed his/her operations. The arising waiting times may significantly increase the processing times of the orders, implying that a large number of pickers does not guarantee for small processing times. Therefore, in this paper, the impact of several problem parameters on the amount of waiting time is investigated first and situations are identified where the consideration of waiting times is inevitable for an efficient organization of the picking process. In the second part of the paper, a solution approach, namely a truncated branch-and-bound algorithm, is proposed which aims for the minimization of the waiting times. By means of extensive numerical experiments, it is demonstrated that this approach provides high-quality solutions within a very small amount of computing time.

Keywords: Order Picking, Picker Routing, Picker Blocking

Corresponding author:
André Scholz
Postbox 4120, 39016 Magdeburg, Germany
Phone: +49 391 67 51841
Fax: +49 391 67 48223
Email: andre.scholz@ovgu.de

1 Introduction

Zalando, a large mail order company, recorded an increase of the number of customer orders by more than 400% in recent years, as the number of orders amounted to 11.0 million in 2011, while 55.3 million orders were received in 2015 (Statista, 2016). When placing an order, customers have the possibility to choose express deliveries, guaranteeing the requested items to be delivered at the next work day. Recently, even same-day deliveries have been tested in some regions. Thus, being able to process customer orders very fast becomes more important in order to ensure customer satisfaction. Before the items requested by the customers can be shipped to the customer locations, the orders have to be processed in the distribution center (warehouse), i.e. the items have to be retrieved from their storage locations. In most warehouses, this is done by human operators (order pickers) who perform tours through the warehouse. For processing a huge number of orders within a short amount of time, many order pickers are employed who work in the warehouse at the same time.

Besides a large number of orders, companies are confronted with an increasing number of different articles to be stored (Hirschberg, 2015). Due to limited space, warehouses often include narrow picking aisles in order to maximize space utilization (Gue et al., 2006). However, in narrow aisles, order pickers can neither pass nor overtake each other. When two pickers work in a narrow aisle at the same time, a picker may have to wait until the other picker has completed the work in this aisle. This can cause severe problems, as waiting times may arise on a large scale and the advantage of the employment of a large number of pickers diminishes. Although it is known that waiting times have a significant negative impact on the processing times, waiting times are rarely taken into account in the literature when guiding pickers through the warehouse.

The intention of this paper is twofold. A large variety of analytical and simulation models exists which estimate the impact of several problem parameters on the waiting times. However, almost all approaches rely on the assumption that all storage locations have to be visited regardless of the locations of requested items. In order to provide more realistic insights, we conduct extensive numerical experiments for the evaluation of the impact of the parameters. Combinations of parameters are identified where waiting times are significant and its consideration is inevitable for an efficient organization of the picking process. In the second part of the paper, a solution approach is provided which takes the waiting times into account. In fact, we propose a truncated branch-and-bound algorithm, where waiting instructions are given to order pickers. Such instructions include information about the points in time when a picker has to wait and when he/she continues the tour. By means of this approach, the benefit of using more sophisticated waiting instructions as well as the impact of the decisions regarding the selection of the picker who has to wait are investigated.

The remainder of the paper is organized as follows. A detailed description of the problem is given in the next section. Section 3 comprises a literature review. First, the results obtained by means of analytical and simulation models are reviewed. Second, solution approaches are presented which deal with guiding order pickers through the warehouse while taking waiting times into account. In Section 4, the impact of several parameters on the waiting times is investigated. Since waiting times are significant for several parameter combinations, a truncated branch-and-bound algorithm is proposed which aims for the minimization of the waiting times (Section 5). Section 6 is devoted to the evaluation of the performance of the algorithm. The paper concludes with a summary and an outlook on future research opportunities.

2 Problem description

In manual picker-to-parts order picking systems, order pickers walk or ride through the warehouse in order to retrieve requested items from their storage locations. The storage locations are typically arranged in such a way that they constitute a block layout (Roodbergen, 2001). A picking area following a block layout includes two types of aisles: picking aisles and cross aisles. Picking aisles are of identical length and width and are arranged parallel to each other. Furthermore, they have to be entered to retrieve items as the items are stored on pallets or racks located on one side or even both sides of the picking aisles. Cross aisles are arranged orthogonally to the picking aisles. They do not contain any requested items, but cross aisles are required for enabling the pickers to proceed from one picking aisle to another. Cross aisles divide the picking area into blocks and the picking aisles into subaisles (see Fig. 1).

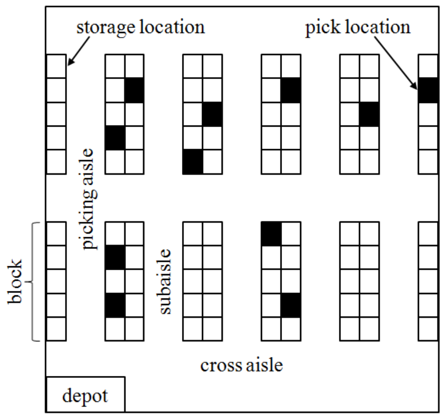


Fig. 1: Two-block layout

In Fig. 1, a two-block layout is depicted which contains 5 picking aisles and 10 subaisles. The rectangles symbolize the storage locations while the locations of requested items (pick locations) are represented by black rectangles. The depot is located in front of the leftmost picking aisle. A picker tour then starts at the depot, proceeds to the respective pick locations and ends at the depot. The time that an order picker needs for performing a tour (processing time) is composed of (Tompkins et al., 2010) the time required for preparing the tour (setup time), the time spent at the pick locations for the identification and the retrieval of the items (pick time) and the time needed for traveling from the depot to the pick locations, between the pick locations and back to the depot (travel time). Since a picking area with narrow subaisles is considered, an additional component, namely the waiting time, has to be taken into account. Waiting times arise because order pickers can neither pass nor overtake each other in narrow subaisles. Thus, several order pickers working in the same subaisle at the same time may cause congestion (blocking). (Note that congestion is not an issue in cross aisles.) A situation where an order picker cannot continue his/her operations because he/she is not able to pass or overtake another picker is referred to as a blocking situation. An example for a blocking situation is depicted in Fig. 2. Here, picker #1 is retrieving an item from its storage location. At the same time, picker #2 has to pass this location in order to reach another pick location. Due to the narrow subaisle, picker #2 is not able to pass picker #1. Thus, picker #2 has to wait until picker #1 has completed the retrieval of the item (assuming that picker #1 will proceed the tour by going upwards).

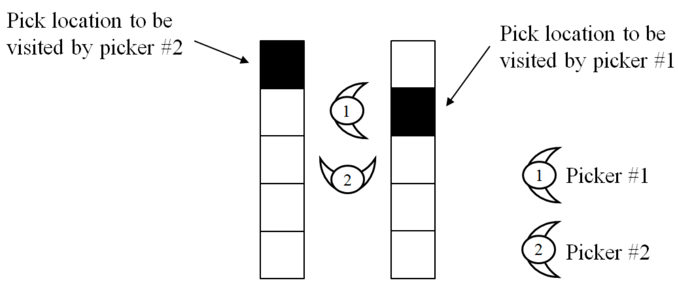


Fig. 2: Two order pickers working in the same subaisle

From the components of the processing time, the setup time and the pick time can be considered as constants (Bozer & Kile, 2008; Henn et al., 2010). The travel time is dependent on the sequence according to which the pick locations are meant to be visited. This sequence is determined by means of a certain procedure here, e.g. by application of a routing strategy (Roodbergen, 2001) or even by using an exact approach (Ratliff & Rosenthal, 1983; Roodbergen & de Koster, 2001). Thus, the sequence and, therefore, also the travel time can be assumed to be known for a given set of requested items, leaving the waiting time as the only variable component of the processing time. The waiting time of an order is dependent on the waiting instructions given to the picker performing the corresponding tour. A waiting instruction may have to be executed when a blocking situation arises. It comprises information about the point in time when a picker starts to wait (which also defines the position in the picking area where the picker waits) and the point in time when the picker continues the tour.

A set of customer orders to be processed is given. Each customer order is specified by the date when it has become available at the warehouse (arrival date) and by the requested articles and the respective quantities. The orders are processed by the pickers in the sequence they arrived at the warehouse (first-come-first-served) while a separate tour is performed for processing each customer order. As soon as an order picker becomes available, i.e. when he/she has finished a tour, he/she immediately starts with processing the next order in the sequence.

It is of prime importance to process customer orders as fast as possible. Therefore, the minimization of the throughput time of all orders (total throughput time) is a very common objective in this context (Le-Duc & de Koster, 2007; Van Nieuwenhuyse & de Koster, 2009; Yu & de Koster, 2009). The throughput time of an order is defined as the difference between the completion date of the order, i.e. the point in time when all requested items have been brought to the depot, and its arrival date. The throughput time of an order is composed of the time that elapsed after its arrival until processing of the order has started (start date) and its processing time. The start date of an order cannot be affected directly, as the sequence is given according to which the orders are processed. Indirectly, it is affected by the processing times of the orders processed before. Concerning the processing time of an order, as mentioned before, the waiting time is the only variable part. Thus, the minimization of the waiting times of all orders (total waiting time) is equivalent to the minimization of the total throughput time here.

The problem can now be stated as follows. Let a set of customer orders with known arrival dates be given including certain requested items. The customer orders are processed by a certain number of order pickers according to the sequence in which they arrived. Each picker processes the next order as soon as the picker becomes available. Furthermore, let setup times, pick times per item as well as a constant travel velocity of the pickers be given. In addition, the layout of the picking area is known and a routing algorithm for the construction of the picker tours is given. Then, for each order picker, the points in time when the picker has to wait and when he/she has to continue the tour have to be determined, respectively, in such a way that the total waiting time is minimized.

In the following section, the related literature is reviewed. First, we focus on analyses of the impact of several problem parameters on different performance criteria regarding the efficiency of the picking process. Second, solution approaches to related problems are reviewed.

3 Literature review

3.1 Analyses regarding the impact of parameters

A general consensus has been reached in the literature regarding the point that picker blocking may have a significant negative impact on the efficiency of the picking process. Thus, it is not surprising that a large variety of approaches exists concerning the estimation of the impact of certain problem parameters on performance criteria related to the picking process such as the total throughput time or the proportion of the total waiting time as part of the processing time of all orders (total processing time). All approaches deal with analytical and simulation models where most of them are based on the assumption that all subaisles are traversed according to a given sequence and direction (regardless of the fact whether a subaisle includes a pick location or not). It can then be assumed that the storage locations constitute a cycle. The order pickers start at a certain point of the cycle which represents the depot. From this point, they walk through the cycle until they reach this point again. With a probability of p (referred to as the pick density), an order picker stops at a storage location in order to retrieve an item.

Parikh & Meller (2009) dealt with picker blocking arising in warehouses with wide aisles, i.e. order pickers are able to pass and overtake each other in all aisles. However, pickers may block each other when the same pick location has to be visited at the same time (pick-face blocking). The authors pointed out that the proportion of the waiting time increases with an increasing pick density p. When p exceeds a certain value, waiting times decrease with a further increasing p. If p is equal to 1, no waiting times will arise as the pickers will stop at each location, implying that all pickers need the same time for performing a tour through the cycle. Furthermore, Parikh & Meller (2009) observed that a larger number of storage locations results in shorter waiting times, whereas the proportion of the total waiting time significantly increases when a larger number of pickers is available.

Skufca (2005) considered the impact of the number of order pickers, the number of storage locations and the pick density on the proportion of the total waiting time as part of the total processing time. The author dealt with a narrow-aisle warehouse, i.e. waiting times may arise since passing and overtaking of order pickers is not possible in subaisles (in-the-aisle blocking). Regarding the impact of the parameters mentioned above, Skufca (2005) obtained the same results as Parikh & Meller (2009). Based on the same assumptions, Gue et al. (2006) investigated the impact of the pick density but also of the pick-walk-time ratio, i.e. the average pick time per item divided by the time required for passing a storage location without retrieving an item. Gue et al. (2006) observed that an increasing pick-walk-time ratio leads to an increasing proportion of the total waiting time. Parikh & Meller (2010) additionally pointed out that waiting times may be underestimated by far if deterministic pick times per item are assumed. For example, no waiting times occur if the pick density equals 1 and pick times are deterministic. This is not true in case of non-deterministic pick times.

Pan & Shih (2008) and Pan & Wu (2012) are the only publications in which the picking area is not assumed to be cyclic as picker tours through the narrow-aisle warehouse are constructed by means of certain routing strategies. Pan & Shih (2008) applied the S-shape strategy. According to this strategy, each subaisle containing at least one requested item is traversed. An exception may occur in the last subaisle of a block where the picker returns after having retrieved all items in this aisle if this leads to a shorter tour. Pan & Shih (2008) investigated the impact of the procedure according to which articles are assigned to storage locations (storage assignment policy) on the throughput rate. The throughput rate is defined as the number of items retrieved within a certain amount of time. They compared a random storage assignment policy to a storage assignment policy of Jarvis & McDowell (1991) which is based on the demand frequency of the articles. Pan & Shih (2008) observed that application of the random assignment policy results in higher throughput rates. Pan & Wu (2012) chose the total throughput time as the performance criterion and extended the considerations of Pan & Shih (2008) to further routing strategies and several class-based storage assignment policies. They pointed out that the routing strategy leading to the shortest tours in combination with the across-aisle storage assignment policy (Petersen & Schmenner, 1999) results in the smallest total throughput time.

3.2 Solution approaches to related problems

Although the impact of waiting times on the performance of the picking process has widely been studied and observed to be significant in many cases, only few solution approaches exist which actually take waiting times into account when guiding order pickers through narrow-aisle warehouses. In fact, two approaches are available which address problems related to the one described in Section 2.

The scenario considered by Chen et al. (2013) differs from the problem defined in Section 2 regarding three aspects. First, the number of order pickers is restricted to two. Second, the next customer orders are not processed before both pickers have finished their tours. Third, no routing algorithm is given. Chen et al. (2013) proposed an ant colony optimization (ACO) approach to the resulting problem. By means of the ACO algorithm, a tour is constructed for the picker who leaves the depot first. This tour will remain unchanged. The ACO is then used for the determination of the tour of the other picker. The construction of the tour is based on the logical distance between pick locations, which is composed of the travel time between the locations as well as the waiting time caused by the tour of the first picker. Solving an instance with 2 customer orders containing up to 30 items requires 10 seconds of computing time. However, in terms of solution quality, the performance of this approach is hardly better than the performance of a modified S-shape strategy.

Chen et al. (2016) extended the considerations of Chen et al. (2013) to the case of an arbitrary number of pickers. They also designed an ACO approach to tackle this problem. First, the ACO algorithm is applied to construct the tours for all pickers without taking waiting times into account. Thus, as it is the case for the problem described in Section 2, a routing algorithm is given by which tours are determined beforehand. In a second step, blocking situations are identified. If a blocking situation is caused by two order pickers performing picking operations in the same subaisle, then the order picker who enters the subaisle first will perform the operations while the other picker waits at the entrance of this subaisle until he/she can execute the operations without being blocked. If two pickers block each other and at least one of the pickers traverses the subaisle without retrieving items, then it is checked whether the total throughput time can be decreased by traversing another subaisle, i.e. tours are allowed to be altered in the settings of Chen et al. (2016). The authors applied their approach to instances with 10 pickers and 30 requested items per order. Computing times have not been reported. The algorithm does not lead to convincing results concerning the solution quality as solutions provided by simple modifications of the S-shape and the largest gap strategy (Hall, 1993) cannot be improved significantly.

4 Evaluation of the impact of parameters on waiting times

4.1 Test instances

In the literature, several problem parameters have been identified which have an impact on the efficiency of the picking process in narrow-aisle warehouses (see Subsection 3.1). Since most approaches rely on the assumption that all subaisles are visited regardless of the pick locations, we conducted extensive numerical experiments in order to investigate the impact of the parameters on the performance of the picking process for more realistic settings. For the evaluation of the performance, the proportion of the total waiting time as part of the total processing time has been used as done by Skufca (2005), Gue et al. (2006) and Parikh & Meller (2009). Based on the observations from the literature, the impact of the following parameters is analyzed: the number of blocks, the number of picking aisles, the number of pickers, the pick-walk-time ratio, the number of items per order, the storage assignment policy, and the routing algorithm.

In the experiments, the picking area follows a block layout with $b \in \{1,2,3\}$ blocks and $m \in \{5,10\}$ picking aisles. Each subaisle contains 25 storage locations on each side, respectively. The distance between adjacent storage locations amounts to 1 length unit (LU). The same distance has to be covered for entering or leaving a subaisle. The distance between two adjacent picking aisles equals 5 LUs while 1.5 LUs are covered for traveling from the depot to the leftmost picking aisle (Henn & Wäscher, 2012).

Instances with 100 customer orders are considered. The number of requested items per order is uniformly distributed between n_l and n_u with $(n_l, n_u) \in \{(5,25),(10,50)\}$. For the assignment of articles to storage locations, two different procedures are applied, namely the random assignment policy ($a = r$) and the class-based assignment policy ($a = c$) used by Henn & Wäscher (2012). According to the random assignment policy, each storage location has the same probability of being a pick location. In the class-based assignment policy, articles are divided into three classes A, B and C based on the demand frequency. Class A articles are 10% of the articles with the highest demand and account for 52% of the total demand. 30% of all articles are assigned to class B where these articles represent 36% of the total demand. The remaining articles belong to class C and are characterized by quite low demand frequencies. Based on the class, articles are assigned to subaisles. Class A articles are located in 10% of the subaisles nearest to the depot while articles assigned to class C are situated in 60% of the subaisles farthest from the depot. The remaining subaisles include articles from class B. Each article is randomly assigned to a storage location of the corresponding subaisles.

For processing the customer orders, $k \in \{2,3,5,7\}$ order pickers are available. The time that an order picker needs to perform the tasks (see Section 2) is set as follows (Henn, 2015). The setup time amounts to 180 seconds while the picker needs 3 seconds to cover 1 LU. Since the pick-walk-time ratio α is usually 20 or less in practical applications (Gue et al., 2006), $\alpha \in \{3,10,20\}$ is chosen. This implies that a picker needs 9, 30 or 60 seconds for searching and retrieving an item. The picker tours are generated by means of two routing algorithms, namely the S-shape strategy and the Lin-Kernighan-Helsgaun (LKH) heuristic of Helsgaun (2000). The S-shape strategy represents the routing strategy most frequently used in practice (Roodbergen, 2001), while the LKH heuristic leads to very short tours (Theys et al., 2010).

The combination of all parameters mentioned above results in 576 problem classes. For each class, 48 instances have been generated, leading to 27648 instances in total.

4.2 Test solution approaches

For the determination of the proportion of the total waiting time as part of the total processing time, the problem described in Section 2 has to be solved. Two solution approaches are considered which are adapted from approaches proposed in the literature. In both approaches, based on the given picker tours, blocking situations are identified and waiting instructions are given. More precisely, the approaches work as follows. Let k denote the number of order pickers. In the first step, based on their arrival dates, the first k customer orders are processed by the pickers where each picker processes exactly one order. It is then checked whether blocking situations have to be dealt with. Blocking situations are identified chronologically, i.e. the situation which occurs first is considered. A blocking situation always concerns two order pickers. In order to deal with a blocking situation, waiting instructions are given to one of the pickers. The waiting instructions will not change the tour but they may affect the points in time when an order picker is at a certain location. Thus, these points in time have to be updated. Then, the next blocking situation is identified and dealt with based on the updated points in time. When all blocking situations have been considered which arise until one of the pickers has finished his/her current tour, the next customer order is assigned to this picker, and it is again checked whether new blocking situations arise. The procedure is repeated until all customer orders have been assigned to the pickers and all blocking situations have been dealt with. This principle is the same for both approaches presented below. However, the approaches differ with respect to the waiting instructions given to the pickers.

The first approach (A_1) is based on an approach of Ho & Chien (2006). They considered a distribution center in Taiwan where a single order picker was allowed to be in a subaisle only. Thus, an order picker is only permitted to enter a subaisle if no other picker is currently working in this subaisle. Otherwise, the picker has to wait at the entrance of the subaisle until the other picker has left the subaisle. Based on this rule, waiting instructions are given, i.e. the points in time when an order picker has to wait and when he/she has to proceed the tour are determined.

When applying A_1, waiting times can be expected to be very large, as pickers may have to wait although they would not actually block each other according to the definition of a blocking situation (see Section 2). Therefore, in the second approach (A_2), several order pickers are allowed to be in the same subaisle at the same time. In this approach, a blocking situation is dealt with by giving waiting instructions to the picker who left the depot at a later point in time (Chen et al., 2013). Thus, it is known which picker is allowed to continue the tour and which picker has to wait. Waiting instructions for a blocking situation can then be given in such a way that the waiting time caused by this situation is minimized. For the identification of such waiting instructions, the possible paths through subaisles are considered. A picker either traverses a subaisle or returns at a certain point. Moreover, either two pickers enter a subaisle from the same cross aisle or they use different cross aisles. Based on these observations, six blocking situations have to be considered (see Fig. 3).

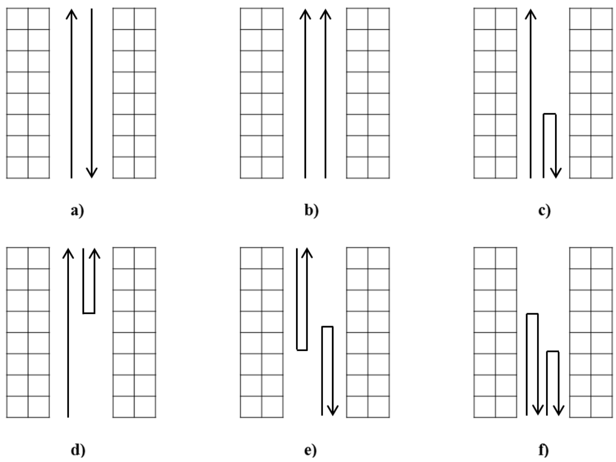


Fig. 3: Possible blocking situations in a subaisle

Waiting instructions are now given to the picker who has to wait based on the classification of the blocking situation. If the blocking situation follows a scenario depicted in Fig. 3 a) to e), the picker waits at the entrance of the subaisle until he/she can proceed the tour without blocking the other picker. The scenario shown in Fig. 3 f) is the only scenario where a picker may wait in the subaisle. This depends on the locations of the return points. If the location of the return point of the picker who has to wait is closer to the cross aisle from where the subaisle has been entered, then the picker will wait at the entrance as done in the other scenarios. If the return location is farther away and if the picker has entered the subaisle first, then he/she may wait at the return location until he/she can proceed the tour without being blocked by the other picker.

4.3 Results

The results of the experiments are depicted in Tables 1 and 2 and Tables A1 to A6, where Tables denoted by an "A" are included in the appendix available at `http://www.mansci.ovgu.de/mansci/en/Research/Materials/2017+_+I_-p-632.html`. Tables 1, 2, A1 and A2 contain information about the average proportion of the total waiting time as part of the total processing time (in %) for the approaches A_1 and A_2 combined with the S-shape strategy and the LKH heuristic, respectively. The corresponding total processing times (in hours) are shown in Tables A3 to A6.

Concerning the routing algorithms, the results are very similar. Both algorithms result in the smallest

Table 1: Proportion [%] of the total waiting time as part of the total processing time for A_1 and the S-shape strategy

α	(n_l, n_u)	b	m	$k=2$		$k=3$		$k=5$		$k=7$	
				$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$
3	(5,25)	1	5	1.6	2.3	3.5	5.2	8.0	13.0	15.4	28.6
3	(5,25)	1	10	1.1	1.9	2.2	3.9	4.6	9.5	7.4	18.4
3	(5,25)	2	5	1.4	2.5	3.1	5.3	6.4	13.0	10.1	24.4
3	(5,25)	2	10	0.9	1.3	2.0	2.9	4.0	6.9	6.1	11.9
3	(5,25)	3	5	1.0	1.8	2.3	3.6	4.8	8.2	7.8	13.1
3	(5,25)	3	10	0.6	1.1	1.2	2.2	2.7	5.0	4.4	8.4
3	(10,50)	1	5	2.0	2.9	4.3	6.3	9.8	17.8	20.5	37.1
3	(10,50)	1	10	0.9	2.0	1.9	4.6	4.2	10.9	6.6	23.3
3	(10,50)	2	5	1.4	2.9	2.9	6.0	6.0	14.5	9.3	29.1
3	(10,50)	2	10	0.8	1.3	1.7	2.7	3.5	6.4	5.6	10.8
3	(10,50)	3	5	1.0	1.6	2.1	3.4	4.3	7.6	6.9	12.4
3	(10,50)	3	10	0.6	1.0	1.1	2.0	2.5	4.4	4.3	7.4
10	(5,25)	1	5	2.7	4.4	6.0	10.1	13.3	29.0	23.1	47.6
10	(5,25)	1	10	1.4	3.5	3.1	7.8	6.3	21.3	10.2	40.5
10	(5,25)	2	5	1.8	4.1	3.8	9.6	8.1	26.0	12.6	44.3
10	(5,25)	2	10	1.2	2.1	2.1	4.7	4.4	10.7	7.1	18.5
10	(5,25)	3	5	1.4	2.4	2.8	5.3	5.9	12.1	9.2	20.2
10	(5,25)	3	10	0.7	1.5	1.4	3.2	3.0	7.1	4.9	11.6
10	(10,50)	1	5	4.1	6.0	8.2	14.0	16.7	38.4	27.3	55.3
10	(10,50)	1	10	1.7	4.5	3.6	10.1	7.7	29.3	11.7	48.5
10	(10,50)	2	5	2.2	5.5	4.5	11.8	9.4	33.2	14.0	51.0
10	(10,50)	2	10	1.0	2.3	2.1	4.9	4.6	11.7	7.1	19.6
10	(10,50)	3	5	1.3	2.9	2.9	5.7	6.3	12.7	9.4	21.1
10	(10,50)	3	10	0.7	1.6	1.5	3.2	3.3	7.1	4.9	11.7
20	(5,25)	1	5	4.1	6.8	8.2	16.6	18.0	42.0	28.7	57.6
20	(5,25)	1	10	1.9	5.4	3.9	12.7	8.2	34.8	13.2	52.6
20	(5,25)	2	5	2.5	6.4	4.9	14.6	9.9	37.3	15.4	54.6
20	(5,25)	2	10	1.2	3.1	2.4	6.4	5.1	14.5	8.3	25.0
20	(5,25)	3	5	1.6	3.4	3.3	7.4	7.2	16.2	10.9	26.8
20	(5,25)	3	10	0.8	2.0	1.7	4.1	3.6	9.6	5.6	15.5
20	(10,50)	1	5	5.8	8.8	10.8	20.2	21.0	47.0	32.1	61.7
20	(10,50)	1	10	2.6	6.7	5.4	16.2	10.5	42.1	15.6	58.1
20	(10,50)	2	5	2.9	7.5	5.5	18.0	11.5	43.7	17.5	59.2
20	(10,50)	2	10	1.4	3.2	2.6	7.0	5.6	16.3	8.4	28.0
20	(10,50)	3	5	1.9	3.8	3.8	7.9	8.0	17.0	11.7	28.4
20	(10,50)	3	10	0.9	2.3	1.8	4.6	3.8	9.5	5.9	16.0

proportion of the total waiting time for the problem class ($\alpha = 3$, $(n_l, n_u) = (5, 25)$, $b = 3$, $m = 10$, $k = 2$, $a = r$). When applying A_1 the smallest proportion of the waiting time amounts to 0.6% for both routing algorithms, while the proportion equals 0.4% and 0.5% for the S-shape strategy and the LKH heuristic when A_2 is used. For the S-shape strategy, the maximum proportion of the waiting time is 61.7% for A_1 and 22.2% for A_2. Regarding the LKH heuristic, proportions of up to 62.2% and 28.6% can be observed. On average, order pickers wait for 10.9% or 4.7% of the total processing time if A_1 or A_2 is applied and tours are constructed by means of the S-shape strategy. When using the LKH heuristic, waiting times account for 12.3% or 6.0% of the total processing time. It can be seen that the proportions are slightly

Table 2: Proportion [%] of the total waiting time as part of the total processing time for A_2 and the S-shape strategy

α	(n_l, n_u)	b	m	$k=2$		$k=3$		$k=5$		$k=7$	
				$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$
3	(5,25)	1	5	1.3	0.9	2.9	2.0	6.1	3.9	10.0	6.1
3	(5,25)	1	10	1.0	0.7	1.8	1.3	3.7	2.8	5.7	4.5
3	(5,25)	2	5	1.1	1.0	2.1	2.0	4.6	4.3	6.7	7.0
3	(5,25)	2	10	0.6	0.9	1.4	2.0	2.7	4.1	4.3	6.4
3	(5,25)	3	5	0.7	0.8	1.5	1.6	3.2	3.7	4.9	5.6
3	(5,25)	3	10	0.4	0.6	0.8	1.3	1.7	2.8	2.8	4.3
3	(10,50)	1	5	1.7	1.1	3.6	2.4	7.7	4.6	13.8	6.9
3	(10,50)	1	10	0.7	0.6	1.5	1.3	3.1	2.7	4.7	4.2
3	(10,50)	2	5	1.2	1.4	2.4	2.7	5.0	5.7	7.3	8.8
3	(10,50)	2	10	0.7	1.0	1.3	2.0	2.8	4.4	4.4	6.8
3	(10,50)	3	5	0.8	0.9	1.8	1.9	3.4	3.9	5.5	6.1
3	(10,50)	3	10	0.5	0.6	0.9	1.1	1.9	2.4	3.1	3.7
10	(5,25)	1	5	2.3	1.7	4.5	3.4	9.6	6.6	15.0	10.4
10	(5,25)	1	10	1.2	1.2	2.7	2.4	5.3	4.9	8.0	7.8
10	(5,25)	2	5	1.4	1.4	2.8	3.2	6.0	6.4	8.9	10.0
10	(5,25)	2	10	0.9	1.5	1.6	2.9	3.5	6.1	5.5	9.5
10	(5,25)	3	5	1.0	1.2	2.1	2.4	4.2	5.3	6.5	8.3
10	(5,25)	3	10	0.5	1.0	1.0	1.9	2.1	4.1	3.4	6.2
10	(10,50)	1	5	3.1	2.2	6.2	4.2	12.2	8.2	18.7	11.9
10	(10,50)	1	10	1.4	1.3	2.8	2.7	5.8	5.2	8.4	7.9
10	(10,50)	2	5	1.8	2.1	3.4	4.0	7.1	8.0	10.7	12.1
10	(10,50)	2	10	0.9	1.7	1.9	3.5	3.9	7.1	5.9	11.3
10	(10,50)	3	5	1.1	1.4	2.5	2.6	5.1	5.3	7.7	8.2
10	(10,50)	3	10	0.6	0.8	1.3	1.7	2.7	3.8	4.0	5.7
20	(5,25)	1	5	3.0	2.7	6.2	5.1	12.5	9.8	18.8	14.3
20	(5,25)	1	10	1.8	1.8	3.4	3.5	7.0	7.2	10.4	11.3
20	(5,25)	2	5	1.9	2.2	3.7	4.5	7.4	8.8	10.9	13.7
20	(5,25)	2	10	1.1	1.9	2.0	3.9	4.2	8.4	6.6	13.1
20	(5,25)	3	5	1.3	1.8	2.6	3.4	5.3	7.2	7.8	10.9
20	(5,25)	3	10	0.7	1.3	1.3	2.6	2.7	5.9	4.3	8.7
20	(10,50)	1	5	4.3	3.0	7.7	6.2	14.8	10.7	22.2	14.9
20	(10,50)	1	10	2.1	2.0	4.2	3.8	7.9	7.6	11.6	11.5
20	(10,50)	2	5	2.4	2.6	4.6	5.4	9.2	10.2	13.5	15.0
20	(10,50)	2	10	1.3	2.0	2.5	4.4	4.9	9.4	7.3	14.5
20	(10,50)	3	5	1.6	1.7	3.1	3.4	6.3	6.8	9.1	10.5
20	(10,50)	3	10	0.7	1.2	1.6	2.4	3.3	5.0	5.2	7.7

larger if tours are generated by application of the LKH heuristic. This can be explained by the fact that the LKH heuristic constructs shorter tours, resulting in smaller total processing times (see Tables A3 to A6). The impact of the other parameters is nearly the same for both routing algorithms. Therefore, the analysis is based on the results related to the S-shape strategy only.

Number of order pickers

According to the literature, waiting times significantly increase with a rising number of order pickers k since more blocking situations arise when many pickers work in the same picking area at the same time.

This is also verified by the results of the experiments. While the average proportion of the total waiting time amounts to 2.5% and 1.4% for A_1 and A_2 when the number of order pickers is very small ($k = 2$), the pickers spend 21.1% and 8.7% of their time on waiting for other pickers performing their operations, respectively, if many pickers are simultaneously employed ($k = 7$).

Size of the picking area and storage assignment policy

The size of the picking area is dependent on the number of blocks b and the number of picking aisles m. If the picking area is quite large, order pickers do not come across each other very often, resulting in few blocking situations and a short total waiting time. The same line of argumentation holds for the application of the random assignment policy instead of using the class-based storage assignment strategy. Thus, it can be expected that the proportion of the total waiting time decreases with increasing values for b and m, and that the proportion is smaller for the random assignment policy. In Table 3, the average proportion of the total waiting time is depicted for A_1 and A_2 dependent on the number of blocks, the number of picking aisles and the storage assignment policy.

Table 3: Proportion [%] of the total waiting time dependent on the size of the warehouse and the storage assignment policy

b	m	A_1		A_2	
		$a = r$	$a = c$	$a = r$	$a = c$
1	5	12.3	24.1	8.7	6.0
1	10	5.7	19.5	4.4	4.2
2	5	7.0	21.9	5.3	5.9
2	10	3.7	9.3	3.0	5.4
3	5	4.9	10.2	3.7	4.4
3	10	2.6	5.9	2.0	3.2

As can be seen in Table 3, the results of the experiments match with the expectations if A_1 is applied. The proportion of the total waiting time decreases with an increasing number of blocks (15.4% for $b = 1$, 10.4% for $b = 2$ and 5.9% for $b = 3$), it decreases with a rising number of picking aisles (13.4% for $m = 5$ and 7.8% for $m = 10$) and the proportion gets smaller when the random assignment policy is applied (15.1% for $a = c$ and 6.0% for $a = r$).

Concerning A_2, the impact of the size of the picking area is dependent on the storage assignment policy. For the random assignment policy ($a = r$), the proportions of the total waiting time are in line with the expectations as they decrease with increasing numbers of blocks (5.8% for $b = 1$, 4.9% for $b = 2$ and 3.3% for $b = 3$) and picking aisles (5.7% for $m = 5$ and 3.7% for $m = 10$). However, if the class-based assignment procedure is used ($a = c$) and the picking area contains 10 picking aisles, the proportions increase when switching from 1 block to 2 blocks. Furthermore, the proportion of the waiting time is larger for ($b = 3$, $m = 5$) than for ($b = 1$, $m = 10$), although the latter picking area contains fewer subaisles. This can be explained by the procedure how the articles are assigned to the three classes A, B and C. The articles included in classes A and B account for 88% of the total demand and they are distributed over the 40% of the subaisles which are nearest to the depot. In case of a two-block layout with 5 picking aisles, classes A and B are solely assigned to subaisles of the block nearest to the depot (first block). Thus, it is quite likely that all pick locations included in a tour are situated in the first block. Moreover, most of the subaisles assigned to classes A and B will have to be visited. Tours constructed by means of the S-shape strategy are then very similar as all of these subaisles are traversed. This results in quite short waiting times since order pickers may only be blocked by other pickers who currently retrieve an item. In contrast, if picking areas include 2 blocks and 10 picking aisles or 3 blocks, then subaisles of the second block are assigned to class B as well. Therefore, at least two blocks are part of the tours, which makes the resulting tours much more diverse. Order pickers then traverse subaisles in different directions. If an order picker is blocked by another picker who traverses the subaisle in a different direction, the picker has to wait until the other picker has left the subaisle. In most cases, this causes considerably larger waiting times than blocking situations where both pickers traverse an aisle in the same direction. Thus, the proportion of the total waiting time increases if not all frequently requested articles are assigned to subaisles of the first block.

Pick-walk-time ratio

In the experiments, the travel velocity of the pickers has been fixed and the time required for performing the operations at a pick location is varied. The larger the pick-walk-time ratio α gets, the longer an order picker stops at a pick location. Thus, it can be expected that a larger value for α leads to an increasing proportion of the total waiting time (Gue et al., 2006). The results of the experiments match with the expectations. For the application of A_1, the average proportion amounts to 6.5% for $\alpha = 3$, to 11.0% for $\alpha = 10$ and to 14.3% for $\alpha = 20$ while the average proportions equal 3.1%, 4.7% and 6.2%, respectively, if A_2 is used.

Number of requested items per customer order

According to Gue et al. (2006), waiting times increase with an increasing number of requested items per order. Furthermore, Hong et al. (2010) pointed out that a larger variance in the number of items will increase the proportion of the waiting time. Thus, it is expected that larger proportions can be observed for classes with $(n_l, n_u) = (10, 50)$. This is true for both A_1 and A_2 as the average proportions rise from 9.9% and 4.4% to 11.3% and 4.9%, respectively, when (n_l, n_u) are raised from $(5, 25)$ to $(10, 50)$.

Besides the impact of different problem parameters on the proportion of the total waiting time, the results of the experiments clearly show that, in many settings, taking waiting times into account is pivotal for achieving small processing times. In some settings, more than half of the total processing time can be attributed to waiting times, i.e. order pickers spend more time on waiting than on traveling through the warehouse and retrieving items. In order to keep waiting times at a reasonable level, a truncated branch-and-bound algorithm is presented in the next section.

5 A truncated branch-and-bound algorithm

5.1 General overview

In both solution approaches presented in Subsection 4.2, waiting instructions are given to a predefined picker (e.g. the picker who enters a subaisle at a later point in time). This may result in high waiting times. For example, if a picker enters a subaisle first but has to pick many items in this subaisle, then a picker, whose current tour only includes few pick locations in this subaisle, may have to wait for a long time. Therefore, a solution approach is presented which deals with the determination of the picker to whom waiting instructions are given. Since exactly two decisions are possible in this case, it seems reasonable to apply a branch-and-bound algorithm. Due to computing time and memory issues, a truncated branch-and-bound (TBB) algorithm has been designed. In TBB algorithms, the branching scheme of a branch-and-bound algorithm is kept while heuristic evaluation methods are applied to prune some branches (Rakrouki et al., 2012). By the heuristic pruning of branches, the computational effort is considerably reduced. However, optimality of the solution obtained cannot be guaranteed.

A pseudo-code of the TBB algorithm designed here is depicted below. In the TBB algorithm, each node of the tree represents a partial solution. At the beginning of the algorithm, no assignments of orders to pickers have been performed. The root r is then constructed by application of the expansion procedure. In the expansion procedure, (some) customer orders are assigned to order pickers and blocking situations are identified which arise by processing the orders according to the tours constructed by the given routing algorithm. After the expansion procedure is completed, the root is either assigned to the set of active nodes V or to the set of terminal solutions F. A node represents a terminal solution if all customer orders have been assigned to order pickers and all blocking situations have been taken into account. If r already corresponds to a terminal solution, the TBB algorithm terminates. Otherwise, iterations are performed as long as active nodes exist. An iteration starts with the selection of an active node $\tilde{v}$. A branching procedure is then applied to $\tilde{v}$, resulting in two nodes v_1 and v_2 located on the next level of the tree. By branching, the decision about the picker who has to wait is taken for a certain blocking situation and waiting instructions are given to this picker. The expansion procedure is applied to v_1 and v_2, respectively. It is then checked whether the nodes represent a terminal solution or have to be included in the set of active nodes. At the end of an iteration, the pruning procedure identifies active nodes which are excluded from the solution process, i.e. they are removed from the set of active nodes. At the end of the solution process, the TBB algorithm returns the node v^* which corresponds to the terminal solution resulting in the minimum total waiting time w. In the following, the components of the TBB algorithm are explained in greater detail.

Algorithm 1 Truncated Branch-and-Bound Algorithm

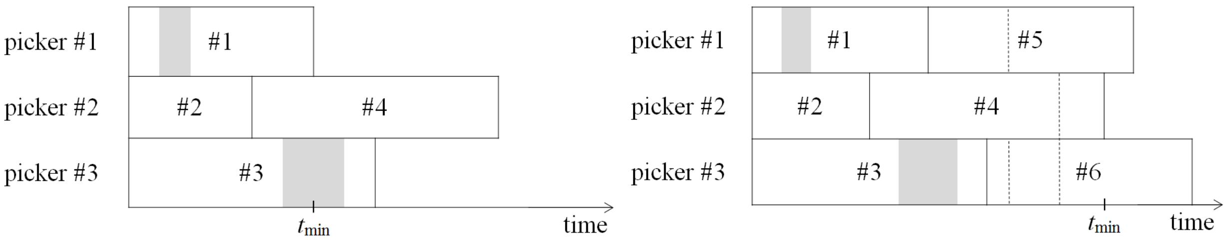
Input: problem data, node r corresponding to a partial solution with no orders assigned to pickers;

Output: node v^* corresponding to a solution to the problem defined in Section 2;

```
r := Expansion_Procedure(r);
if r corresponds to a terminal solution then
    V := ∅;  F := {r};
else
    V := {r};  F := ∅;
end if
while V ≠ ∅ do
    ṽ := Node_Selection(V);  V := V \ {ṽ};
    (v1, v2) := Branching_Procedure(ṽ);
    for v ∈ {v1, v2} do
        v := Expansion_Procedure(v);
        if v corresponds to a terminal solution then
            F := F ∪ {v};
        else
            V := V ∪ {v};
        end if
    end for
    for v ∈ V do
        Pruning_Procedure(v);
    end for
end while
v* := arg min {w(v) | v ∈ F};
```

5.2 Expansion of a node

As mentioned before, the TBB algorithm starts with no customer orders being assigned to the order pickers and it then successively assigns the orders to the pickers. This is done in the expansion procedure which works as follows. Let a node v be given, representing a partial solution with a set of customer orders already assigned to order pickers and a set of waiting instructions already given to the pickers. Let t_{min} denote the point in time when the next order picker becomes available where the calculation of t_{min} includes the waiting time of instructions already given but not waiting times which may arise due to blocking situations not yet taken into account. The node v can then be expanded if and only if all blocking situations arising until t_{min} have been dealt with in the corresponding partial solution. In this case, the next order in the sequence (based on the arrival date) is assigned to the picker who finishes the tour at t_{min} and t_{min} is updated. This procedure is repeated until at least one blocking situation arises. At the end of the expansion or if the node cannot be expanded, the expansion procedure returns the blocking situations which have been identified. An example of the expansion of a node is depicted in Fig. 4.



a) Partial solution before expansion of the node

b) Partial solution after expansion of the node

Fig. 4: Example of an expansion of a node

In Fig. 4a), a partial solution is given which corresponds to an expandable node. A Gantt chart is depicted where the rectangles represent the tours to be performed for processing the respective orders. The width of a rectangle gives information about the duration of a tour. The gray parts of a rectangle stand for the waiting time caused by executing the waiting instructions. It can be seen that customer orders #1 to #4 have already been assigned to order pickers in this partial solution. Furthermore, picker #1 and picker #3 execute a waiting instruction, respectively. Here, t_{min} is defined as the point in time when picker #1 has completed the tour. Thus, when expanding the node, the next order is assigned to picker #1 which is shown in Fig. 4b). Since no blocking situation results from this assignment, order #6 is assigned to picker #3 because this picker will be the next picker who is available. This assignment causes two blocking situations which are illustrated by the dotted lines in Fig. 4b). The first blocking situation concerns pickers #1 and #3 while the other situation relates to pickers #2 and #3. Thus, the expansion procedure terminates and t_{min} is now the point in time when picker #2 has processed customer order #4. (Note that the effect of the two blocking situations is not included in t_{min}.)

5.3 Characteristics of a (partial) solution

Let $t_{\min}$ be defined as in the previous subsection. A (partial) solution is then characterized by:

1) the customer orders already assigned to a picker and the assignment of the orders to the pickers;

2) for each picker, the waiting instructions received for the blocking situations already dealt with;

3) the total waiting time caused by performing the received waiting instructions;

4) the number of assigned customer orders;

5) the number of remaining blocking situations arising until $t_{\min}$.

The characteristics mentioned in 1) and 2) contain information about the decisions to be taken for solving the problem described in Section 2. The third component represents the objective function value if the solution is a terminal solution. Otherwise, it defines a lower bound regarding the objective function value. Since the waiting time is dependent on the customer orders and the blocking situations already taken into account, components 4) and 5) are required for the identification of the pairs of the corresponding nodes which can be compared regarding the lower bounds in the pruning procedure. The selection of the node to be considered in an iteration is also based on these components.

5.4 Selection of a node, and branching and pruning procedures

In order to keep the tree at a reasonable size, the node to be considered is chosen in such a way that many nodes can be compared in the pruning procedure, i.e. the corresponding partial solutions of the nodes are equal with respect to the number of assigned customer orders. Therefore, a node is selected according to the following priorities:

1) the smallest number of assigned customer orders;

2) the smallest number of remaining blocking situations;

3) the smallest total waiting time;

4) the first generated node.

The branching procedure is then applied to the selected node. In this procedure, the first arising blocking situation identified in the expansion procedure is considered and two nodes are generated. The generation of the nodes is based on the decision regarding the picker who has to wait in this blocking situation. Waiting instructions are then given to the respective picker. The waiting instructions match with the instructions used in approach A_2 (see Subsection 4.2).

In a TBB algorithm, the pruning procedure replaces the bounding phase of a classic branch-and-bound algorithm. Here, two heuristic methods are applied to prune branches, reducing the size of the tree. The first procedure is based on the comparison of nodes regarding the lower bounds. As mentioned before, nodes can only be compared if they relate to partial solutions characterized by the same number of assigned customer orders. A branch corresponding to a node is then pruned if another node exists whose corresponding partial solution either contains fewer remaining blocking situations while not having a larger total waiting time or if the number of remaining blocking situations is equal for both partial solutions but the total waiting time is smaller for the other one.

The second possibility for pruning a branch of a node is related to the number of remaining blocking situations after application of the branching procedure. By branching, a blocking situation is taken into account and waiting instructions are given to a picker. In general, the number of remaining blocking situations either decreases by 1 or further orders can even be assigned until new blocking situations are identified. However, since the execution of waiting instructions results in changes in the points in time when the respective picker is at certain locations, it is also possible that blocking situations arise which did not occur before. Thus, the number of remaining blocking situations may remain unchanged or even increase. In this case, the branch of the generated node is pruned. An exception occurs if both resulting branches would have been pruned. The node whose corresponding partial solution shows the smaller number of remaining blocking situations is then kept, guaranteeing the algorithm to find a terminal solution.

6 Performance of the TBB algorithm

6.1 Setup

For the evaluation of the performance of the TBB algorithm, numerical experiments are conducted. The settings of the experiments are chosen according to the setup of the experiments described in Subsection 4.1. Based on the observations from Subsection 4.3, we focus on problem classes in which the proportion of the total waiting time as part of the total processing time can be expected to be significant. Thus, the problem classes with the following parameters are considered. For processing $N \in \{100, 200\}$ customer orders, each including between 10 and 50 items, $k \in \{3, 5, 7, 10\}$ order pickers are available while the pick-walk-time ratio α either amounts to 10 or is equal to 20. The picking area of the warehouse contains $b \in \{1,2\}$ blocks and $m \in \{5,10\}$ picking aisles. Articles are assigned to storage locations according to the random or the class-based assignment policy.

The combination of all parameter values gives rise to 256 problem classes. For each class, 48 problem instances have been generated, resulting in 12288 instances in total. The TBB algorithm has been implemented using Visual Studio C++ 2015. The numerical experiments have been executed by means of a Haswell system with up to 3.2 GHz and 16 GB RAM per core.

The performance of the TBB algorithm is evaluated with respect to the amount of improvement (in terms of the reduction of the total waiting time) obtained compared to the application of the approaches A_1 and A_2, i.e. the impact of the decisions regarding the given waiting instructions and the selection of the picker who has to wait is considered. Furthermore, computing times are reported in order to investigate whether the TBB algorithm is able to deal with large-sized instances.

6.2 Improvements obtained by application of the TBB algorithm

6.2.1 Improvements by allowing several pickers to work in the same aisle at the same time

In Tables A7 and A8, the results of the experiments are depicted for problem classes with 100 customer orders where tours have been constructed by means of the S-shape strategy and the LKH heuristic, respectively. Tables 4 and 5 include the respective results for problem classes containing 200 orders. For each problem class, the average total waiting times w_1, w_2 and w_B (in hours) are given which result by the application of the approaches A_1 and A_2 and the TBB algorithm. Furthermore, the average relative amount of reduction of the total waiting time imp_i (in %) is depicted which is obtained by applying the TBB algorithm instead of using approach A_i ($i \in \{1,2\}$).

According to approach A_1, only a single picker is allowed to be in a subaisle (see Subsection 4.2). If a picker has to enter a subaisle currently occupied by another picker, the picker has to wait until the other picker has performed the operations and has left this subaisle. In contrast, more sophisticated waiting instructions are given in the TBB algorithm. Thus, by comparing the total waiting times resulting by application of A_1 and the TBB algorithm, the impact of the waiting instructions on the waiting times can be analyzed. (Note that a further difference between A_1 and the TBB algorithm consists in the selection of the picker who has to wait. However, the impact on the solution quality is quite small compared to the impact of the waiting instructions.)

Table 4: Evaluation of the truncated brach-and-bound algorithm for problem classes with 200 customer orders and S-shape strategy

a	α	b	m	$k=3$					$k=5$					$k=7$					$k=10$				
				w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2
r	10	1	5	8.1	5.8	4.0	50.8	31.5	17.6	12.2	10.7	39.5	12.4	33.1	20.2	16.8	49.2	17.0	70.3	38.2	22.2	68.4	41.9
r	10	1	10	4.3	3.3	1.3	68.7	60.1	8.8	6.3	2.9	66.6	53.4	14.5	10.0	5.1	64.6	48.5	24.1	15.2	8.4	65.1	44.6
r	10	2	5	5.0	4.0	2.2	55.0	44.5	10.4	8.0	5.3	49.1	33.7	16.6	12.4	8.6	47.8	30.2	28.0	19.4	14.3	48.8	26.0
r	10	2	10	3.0	2.7	1.2	60.7	56.0	6.5	5.7	2.7	58.1	51.5	10.3	8.6	4.4	57.0	48.4	16.5	13.2	7.2	56.2	45.4
r	20	1	5	17.3	11.8	7.4	57.2	37.3	36.5	23.9	19.4	46.7	18.7	64.8	38.9	30.5	52.9	21.5	125.3	68.9	39.6	68.4	42.5
r	20	1	10	9.1	7.2	2.8	69.7	61.9	18.7	13.9	6.0	68.1	57.2	28.9	20.4	10.2	64.9	50.3	47.6	30.1	16.4	65.4	45.4
r	20	2	5	9.5	7.2	3.7	61.3	49.0	19.7	15.0	9.1	53.8	39.4	32.1	22.9	15.8	50.9	31.3	52.4	35.1	25.0	52.2	28.7
r	20	2	10	5.3	4.9	2.2	58.2	54.7	11.2	10.1	4.7	57.7	53.0	17.0	15.5	7.5	55.9	51.4	27.1	22.9	12.4	54.2	45.9
average				7.7	5.9	3.1	60.2	49.4	16.2	11.9	7.6	54.9	39.9	27.2	18.6	12.4	55.4	37.3	48.9	30.4	18.2	59.8	40.0
c	10	1	5	13.9	3.7	1.5	89.1	58.8	53.5	7.0	3.5	93.5	50.2	108.3	10.5	6.1	94.4	41.8	188.7	15.9	10.4	94.5	35.0
c	10	1	10	11.1	2.7	1.2	89.6	57.4	41.6	5.4	2.5	94.1	54.9	94.9	8.3	4.2	95.5	49.1	177.4	13.0	7.4	95.8	43.5
c	10	2	5	12.9	4.0	1.9	84.9	51.9	47.1	8.0	4.7	90.1	41.9	100.1	12.5	8.2	91.9	34.9	182.6	20.7	13.3	92.7	35.6
c	10	2	10	6.0	4.2	2.1	64.7	48.6	14.7	8.8	5.4	63.2	38.2	28.1	14.0	9.5	66.1	32.3	64.7	23.6	15.2	76.5	35.7
c	20	1	5	35.7	7.8	3.2	91.1	59.4	122.5	15.2	7.6	93.8	49.7	226.4	22.8	13.1	94.2	42.5	375.8	34.0	21.5	94.3	36.8
c	20	1	10	29.3	6.1	2.5	91.6	59.6	109.1	12.0	5.7	94.8	52.8	211.6	18.8	9.6	95.5	48.8	361.0	27.8	16.5	95.4	40.9
c	20	2	5	31.6	8.0	3.8	88.0	52.6	114.5	16.6	9.4	91.8	43.4	217.2	24.4	15.9	92.7	35.1	368.1	38.9	24.8	93.3	36.2
c	20	2	10	12.5	8.0	4.4	65.2	45.8	31.7	17.2	11.3	64.5	34.7	62.5	28.1	19.8	68.4	29.6	135.0	48.0	28.9	78.6	39.9
average				19.1	5.6	2.6	83.0	54.2	66.8	11.3	6.2	85.7	45.7	131.1	17.4	10.8	87.3	39.2	231.7	27.8	17.2	90.2	37.9

Table 5: Evaluation of the truncated brach-and-bound algorithm for problem classes with 200 customer orders and LKH heuristic

a	α	b	m	$k=3$					$k=5$					$k=7$					$k=10$				
				w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2
r	10	1	5	10.9	9.1	4.7	56.8	48.3	25.4	18.6	11.2	56.0	40.0	42.3	28.8	17.1	59.6	40.5	72.6	45.6	23.4	67.8	48.7
r	10	1	10	6.1	4.4	2.2	63.0	48.8	13.2	8.8	5.0	61.9	43.3	21.2	13.8	8.0	62.3	42.1	35.2	21.2	12.4	64.8	41.3
r	10	2	5	6.5	5.4	2.7	58.7	50.7	13.9	10.7	6.0	56.9	43.9	22.6	16.6	9.6	57.5	42.0	37.2	25.7	14.7	60.3	42.7
r	10	2	10	3.5	2.6	1.3	64.4	51.7	7.6	5.2	2.8	62.7	45.6	11.8	8.0	4.3	63.4	46.4	18.9	12.3	6.9	63.7	44.3
r	20	1	5	20.9	17.2	9.2	55.9	46.3	48.4	35.3	21.3	56.1	39.8	80.8	56.3	32.1	60.3	43.0	133.8	86.8	43.6	67.4	49.7
r	20	1	10	10.9	7.7	4.0	63.6	48.9	23.9	15.7	8.7	63.7	44.9	37.5	24.6	14.3	61.8	41.9	61.6	39.1	21.9	64.4	43.9
r	20	2	5	11.1	9.3	4.7	57.5	49.3	24.4	19.1	10.4	57.6	45.6	39.2	28.8	16.7	57.4	41.9	63.9	44.3	25.4	60.2	42.6
r	20	2	10	5.9	4.3	2.2	63.2	49.4	12.2	8.6	4.6	62.4	46.7	19.3	13.1	7.1	63.2	46.1	30.2	20.2	11.5	62.0	43.1
average				9.5	7.5	3.9	60.4	49.1	21.1	15.2	8.7	59.7	43.7	34.3	23.7	13.7	60.7	43.0	56.7	36.9	20.0	63.8	44.5
c	10	1	5	16.0	3.6	1.9	88.0	46.9	55.6	7.7	4.6	91.7	39.8	110.0	12.3	7.8	92.9	36.8	190.0	20.0	12.2	93.6	38.8
c	10	1	10	13.2	2.8	1.6	87.9	43.4	46.0	5.9	3.4	92.5	42.2	100.2	9.4	5.7	94.3	39.4	181.8	15.1	9.3	94.9	38.1
c	10	2	5	13.8	3.0	1.6	88.6	47.4	50.7	6.4	3.7	92.6	41.8	105.2	10.1	6.5	93.8	35.6	186.0	16.1	10.5	94.4	34.8
c	10	2	10	7.8	6.4	3.3	58.3	49.1	19.7	14.1	8.0	59.4	43.3	35.1	22.4	13.2	62.4	41.0	67.1	36.4	19.2	71.4	47.2
c	20	1	5	37.9	7.5	3.8	89.8	48.8	126.0	15.7	9.7	92.3	38.4	229.3	25.1	15.9	93.0	36.5	379.9	41.2	25.0	93.4	39.4
c	20	1	10	32.2	5.7	3.0	90.7	47.7	114.6	11.7	7.1	93.8	38.9	215.9	19.4	11.8	94.5	39.2	366.9	31.2	19.6	94.7	37.1
c	20	2	5	33.4	6.1	3.2	90.4	47.8	117.6	12.8	7.5	93.6	40.9	219.2	20.1	12.6	94.3	37.4	370.3	32.5	20.3	94.5	37.7
c	20	2	10	15.6	13.7	6.6	57.4	51.3	39.6	28.0	16.7	57.8	40.3	71.2	45.6	26.5	62.8	41.8	133.9	73.8	35.8	73.3	51.5
average				21.2	6.1	3.1	81.4	47.8	71.2	12.8	7.6	84.2	40.7	135.8	20.5	12.5	86.0	38.5	234.5	33.3	19.0	88.8	40.6

By application of the TBB algorithm, significant improvements regarding the total waiting time are achieved. In fact, the improvements range between 39.5% (S-shape, $N = 200$, $k = 5$, $a = r$, $\alpha = 10$, $b = 1$, $m = 5$) and 95.8% (S-shape, $N = 200$, $k = 10$, $a = c$, $\alpha = 10$, $b = 1$, $m = 10$). On average, the total waiting time can be reduced by 72.6%, which corresponds to a reduction of the processing time per customer order by 18 minutes. Thus, the results clearly demonstrate that using appropriate waiting instructions is pivotal for an efficient organization of the picking process.

In the following, the impact of the parameter settings on the amount of reduction obtained by application of the TBB algorithm is investigated. Regarding the number of customer orders and the pick-walk-time ratio, no effect on the amount of reduction can be identified. Concerning the routing algorithms, on average, the improvements obtained are also very similar (72.1% for the S-shape strategy and 73.1% for the LKH heuristic). However, the impact of the remaining parameters may be different depending on how the tours have been constructed.

Tours constructed by means of the S-shape strategy

Concerning the number of order pickers k, the amount of reduction rises with an increasing value of k if the S-shape strategy is used for the construction of the tours and if articles are assigned according to the class-based assignment policy. In fact, the average relative reduction of the total waiting time rises from 82.9% ($k = 3$) to 90.0% ($k = 10$). When assigning articles based on the random assignment policy, the largest improvements are observed in problem classes with 3 or 10 pickers.

Comparing the results for the two storage assignment policies, larger reductions are obtained in each problem class when articles are assigned following the class-based assignment policy. On average, the amount of reduction equals 57.9% and 86.4% for $a = r$ and $a = c$, respectively. If the class-based assignment policy is applied, the subaisles located near to the depot will be visited on almost every tour. Thus, the tours generated by means of the S-shape policy have a very similar structure, implying that order pickers traverse the subaisles in the same direction in most of the tours. In this case, the waiting times can be reduced significantly if other instructions are given than waiting at the entrance of the subaisle until the other picker has left this aisle. Furthermore, based on approach A_1, order pickers may often wait although no blocking situation occurs. If the random assignment policy is used, it is very likely that the sets of subaisles to be visited are significantly different for different tours, increasing the probability of order pickers traversing a subaisle in different directions at the same time. In this case, in both approaches, a picker has to wait until the other picker has left the subaisle. The only possibility of the TBB algorithm to improve the solution consists in the selection of the picker. Thus, the amount of reduction is smaller if the random storage assignment policy is used.

When considering the size of the warehouse for problem classes with $a = c$, the smallest improvements can be observed if the picking area consists of 2 blocks and 10 picking aisles. This can be explained by the fact that this is the only picking area where not all articles from the classes A and B have been assigned to subaisles of the first block. Thus, it is likely that the second block has to be visited, which results to more diverse tours (see also Subsection 4.3). With the same line of argumentation as for the impact of the storage assignment policy on the amount of reduction, smaller amounts of improvement can be justified in this case.

Tours constructed by means of the LKH heuristic

If the tours have been constructed by application of the LKH heuristic, the amount of reduction increases with an increasing number of order pickers for problem classes with $a = c$. This coincides with the observations related to problem classes where the S-shape strategy has been used. The relative reduction of the total waiting time amounts to 81.5%, 83.9%, 85.8% and 88.7% if 3, 5, 7 and 10 pickers are available, respectively. For $a = r$, the relative reduction obtained is similar for problem classes with 3, 5 and 7 pickers, while the largest reductions can be observed in classes with 10 pickers.

Regarding the storage assignment policy, larger relative reductions can be identified for the class-based assignment policy (85.0%) than for the random assignment strategy (61.3%). The only problem classes where the amount of reduction is larger for $a = r$ are characterized by a picking area containing 2 blocks and 10 picking aisles. At the same time, these problem classes represent the classes with the smallest relative reductions obtained if the class-based assignment procedure has been applied. This result matches with the corresponding observation for classes with $a = c$ and tours being constructed by means of the S-shape strategy.

6.2.2 Improvements by selecting the picker who has to wait

The results presented above show that the waiting instructions have a large impact on the total waiting time. In approach A_2 and in the TBB algorithm, identical waiting instructions are given if the same order picker has to wait. The difference between these approaches can only be found in the selection of the picker to whom waiting instructions are given for a certain blocking situation. While the picker who left the depot at a later point in time will always wait according to A_2, the selection of the picker is dependent on the effect of the decision on the current and future blocking situations in the TBB algorithm.

The amount of reduction ranges from 12.4% (S-shape, $N = 200$, $k = 5$, $a = r$, $\alpha = 10$, $b = 1$, $m = 5$) to 61.9% (S-shape, $N = 200$, $k = 3$, $a = r$, $\alpha = 20$, $b = 1$, $m = 10$). As expected, the average relative reduction is smaller than the improvements achieved regarding A_1. However, compared to approach A_2, the TBB algorithm can reduce the total waiting time by 43.2% on average, which shows that the selection of the picker who has to wait also represents an important decision, having a significant impact on the resulting total waiting time.

The reductions obtained for varying numbers of customer orders, pick-walk-time ratios and routing algorithms are of similar magnitude, respectively. Regarding the number of order pickers k, it can be observed that an increasing number of pickers results in smaller relative reductions. While relative reductions of 50.0% are achieved for $k = 3$, the total waiting time can be decreased by 41.1% for $k = 10$. This can be explained by the fact that the total waiting time significantly rises with an increasing number of order pickers. With respect to the absolute reduction of the total waiting time, average improvements of 10.3 hours are obtained for problem classes with 10 pickers, where the waiting time is reduced by 2.4 hours only if 3 pickers are available. The impact of the assignment policy and the size of the picking area is dependent on the underlying routing algorithm. Whereas larger relative reductions can be observed for the class-based assignment policy if tours are constructed by means of the S-shape strategy (42.0% for $a = r$ and 44.1% for $a = c$), the opposite holds for problem classes where the LKH heuristic has been applied (44.9% for $a = r$ and 41.7% for $a = c$). Concerning the size of the warehouse, it can be seen that the number of blocks and the number of picking aisles do not affect the amount of reduction obtained if problem classes based on the LKH heuristic are considered. For classes in which the S-shape strategy has been applied, the average relative reduction drops with a decreasing number of picking aisles if articles have been assigned according to the random assignment policy. Otherwise, the least reductions are obtained in case of a picking area including 2 blocks and 10 picking aisles.

As can be seen from the results of the numerical experiments, both the selection of the order picker who has to wait and the waiting instructions actually given to the respective pickers have a strong impact on the total waiting time. By carefully dealing with both types of decisions, the TBB algorithm manages to reduce the total waiting time significantly, which has a positive effect on the processing times of the orders. In the following subsection, the TBB algorithm is evaluated with respect to the computing time required in order to investigate whether this approach is suitable for dealing with large-sized instances arising in practical applications.

6.3 Computing times

The computing times required by approaches A_1 and A_2 are below one second and, therefore, they are negligible. Regarding the TBB algorithm, for each problem class, the average computing time (in seconds) is depicted in Table 6. As can be seen from the table, the average computing time is below one minute for each problem class. Thus, it can be concluded that the TBB algorithm is suitable for dealing with very large instances as well. More precisely, the average computing time required for solving an instance by means of the TBB algorithm ranges from 0.1 seconds (several classes with S-shape, $N = 100$ and $k = 3$) to 56.9 seconds (S-shape, $N = 200$, $k = 10$, $a = c$, $\alpha = 20$, $b = 2$, $m = 10$).

Table 6: Computing times [seconds] required by the truncated branch-and-bound algorithm

N	a	α	b	m	$k = 3$		$k = 5$		$k = 7$		$k = 10$	
					S-shape	LKH	S-shape	LKH	S-shape	LKH	S-shape	LKH
100	r	10	1	5	0.1	0.2	0.2	0.4	0.5	1.0	2.0	3.7
100	r	10	1	10	0.3	0.3	0.5	0.6	0.9	1.0	1.5	2.2
100	r	10	2	5	0.2	0.3	0.4	0.6	0.7	1.3	1.4	2.7
100	r	10	2	10	0.4	0.5	0.9	0.9	1.7	1.5	3.0	2.6
100	r	20	1	5	0.1	0.2	0.2	0.5	0.6	1.3	2.7	6.2
100	r	20	1	10	0.2	0.3	0.6	0.7	1.0	1.2	2.3	2.4
100	r	20	2	5	0.2	0.3	0.5	0.8	0.8	1.4	1.6	3.4
100	r	20	2	10	0.5	0.5	1.1	1.0	1.9	1.7	3.7	3.2
100	c	10	1	5	0.1	0.1	0.2	0.3	0.3	0.5	0.6	1.1
100	c	10	1	10	0.1	0.2	0.3	0.4	0.4	0.7	0.8	1.2
100	c	10	2	5	0.2	0.2	0.4	0.4	0.8	0.8	2.1	1.4
100	c	10	2	10	0.5	0.6	1.0	1.3	1.8	2.3	5.4	5.8
100	c	20	1	5	0.1	0.1	0.2	0.3	0.4	0.6	0.8	1.6
100	c	20	1	10	0.2	0.2	0.3	0.5	0.7	0.9	1.2	1.5
100	c	20	2	5	0.3	0.2	0.5	0.6	1.0	0.9	2.7	2.0
100	c	20	2	10	0.5	0.7	1.1	1.6	2.5	3.3	8.3	10.1
200	r	10	1	5	0.5	0.8	0.9	2.4	2.2	5.5	8.4	22.7
200	r	10	1	10	0.9	1.3	2.1	2.7	3.7	4.7	6.7	9.2
200	r	10	2	5	1.0	1.4	2.1	3.3	3.4	6.4	6.1	12.9
200	r	10	2	10	1.8	2.2	4.5	4.2	7.5	6.9	14.5	12.4
200	r	20	1	5	0.5	0.9	1.1	2.6	2.5	7.1	10.9	40.2
200	r	20	1	10	1.1	1.3	2.7	3.1	4.8	5.8	9.7	11.2
200	r	20	2	5	1.1	1.5	2.6	3.7	4.2	7.0	8.2	16.8
200	r	20	2	10	2.0	2.3	5.0	4.8	8.6	7.8	17.8	14.4
200	c	10	1	5	0.4	0.5	0.8	1.3	1.3	2.4	2.4	6.1
200	c	10	1	10	0.7	1.0	1.4	2.0	2.2	3.0	3.7	5.9
200	c	10	2	5	1.1	1.0	2.4	2.3	4.4	3.7	9.9	7.2
200	c	10	2	10	2.0	2.7	4.6	5.8	8.6	11.2	26.3	29.1
200	c	20	1	5	0.4	0.6	1.1	1.5	1.7	2.9	3.4	9.7
200	c	20	1	10	0.9	1.1	1.9	2.4	3.1	3.9	5.9	7.7
200	c	20	2	5	1.2	1.2	2.8	2.8	5.4	4.7	15.1	9.3
200	c	20	2	10	2.4	3.2	5.7	7.1	12.6	16.3	45.9	56.9

The number of nodes included in the tree has a major impact on the computing time. Since nodes are generated after a blocking situation has been identified, it can be expected that the largest computing times are required for solving instances from problem classes where order pickers tend to block each other quite often. In fact, computing times rise with an increasing number of order pickers and an increasing pick-walk-time ratio. Furthermore, larger computing times can be observed when tours have been constructed according to the LKH heuristic. This can also be explained by the number of blocking situations arising as, on average, 214 blocking situations are considered for the LKH heuristic while only 188 blocking situations occur for the S-shape strategy. Therefore, if the S-shape strategy has been used for the generation of the tours, the average number of nodes created in the TBB algorithm is lower (1260 nodes for the S-shape strategy compared to 1792 nodes for the LKH heuristic), leading to smaller computing times. The number of customer orders represents another parameter that has an impact on the number of blocking situations. The more customer orders are to be processed, the more tours are to be performed, resulting in a larger number of blocking situations in total. Thus, it is not surprising that computing times increase (from 1.2 seconds to 6.1 seconds) if 200 instead of 100 orders are considered.

A significant part of the computing time is also spent on the identification of blocking situations. Whether a blocking situation arises, is checked each time before a picker enters a subaisle. The more subaisles are to be entered in a tour, the more checks have to be performed. Thus, the identification of blocking situations is more time-consuming in case of large picking areas including many subaisles, leading to slightly higher computing times for increasing numbers of blocks and picking aisles.

7 Conclusion and outlook

In this paper, we dealt with the problem of guiding order pickers through a picking area including narrow subaisles. In narrow subaisles, order pickers can neither pass nor overtake each other. Thus, an order picker may have to wait until another picker has completed the operations in a certain subaisle. Although it is known that, in particular when many order pickers are employed, the arising waiting times have a significant negative impact on the efficiency of the picking process, waiting times are rarely taken into account when guiding order pickers.

In the first part of the paper, by means of numerical experiments, settings are identified where the proportion of the total waiting time as part of the total processing time is quite large and situations are pointed out where waiting times can be neglected. For the determination of the total waiting time, two different approaches are designed in which the decisions regarding the pickers who have to wait are made based on suggestions from the literature. The results of the experiments show that the consideration of waiting times is inevitable for an efficient organization of the picking process, as the proportion of the total waiting time amounts up to 62%, i.e. almost two-thirds of the total processing time is spent on waiting. In order to reduce waiting times, in the second part of the paper, a truncated branch-and-bound algorithm is proposed where blocking situations are identified chronologically and nodes are generated according to decisions regarding the selection of the picker who has to wait in the respective blocking situation. By means of numerical experiments, it is demonstrated that this algorithm leads to excellent results within very short computing times. It is pointed out that waiting times can be decreased by up to 96% if more sophisticated waiting instructions are used instead of instructing the pickers to wait at the entrance of the subaisle until no other picker is in this subaisle. Furthermore, it is shown that reductions of up to 62% can be obtained by simply putting more emphasis on the selection of the picker who has to wait in a certain situation.

It has to be noted that all considerations in this paper are based on the assumption that the travel velocity of all pickers is constant and both the travel velocity and the pick times are deterministic. This is a standard assumption in the literature. However, this assumption is very critical as it is hardly met in practice. First, human operators do not travel with a constant velocity. They have to accelerate after having performed the operations at a location and they decelerate before stopping at a pick location or when switching between picking aisles. Moreover, the travel velocity may differ regarding the travel directions (e.g. the velocity may be lower when an order picker returns to a cross aisle by backing). Second, the travel velocity is not only varying but also stochastic in practice. This also holds for the pick time because a human operator does not need exactly the same amount of time each time he/she performs a certain operation. Thus, the integration of varying or even stochastic travel velocities and pick times represents a very important area of future research.

Further research could also concentrate on the extension of the waiting instructions. In the truncated branch-and-bound approach, pickers either wait or they perform their operations as planned. For the reduction of the waiting times, it could also be advantageous that order pickers deviate from their paths. For example, if a subaisle is traversed without retrieving an item, another subaisle could be chosen (Chen et al., 2016). Moreover, the tours could be modified completely, i.e. the sequence according to which the items are to be picked could be changed. In both scenarios, the minimization of the total waiting time would not represent a valid objective and the total processing time should be used for the evaluation of solutions instead.

The consideration of the assignment of customer orders to the pickers and the sequencing according to which customer orders are to be processed represents another promising topic for future research. These decisions provide much more flexibility, which can be expected to prevent many blocking situations from arising. Another interesting aspect can be found in the integration of the Order Batching Problem, i.e. customer orders can be grouped into batches and then processed on a single tour. It can be expected that the batching of customer orders leads to an increase of the proportion of the total waiting time (Gue et al., 2006; Hong et al., 2010). However, it will significantly reduce the total processing time.

References

Bozer, Y. A. & Kile, J. W. (2008): Order batching in walk-and-pick order picking systems. *International Journal of Production Research* 46, 1887-1909.

Chen, F.; Wang, H.; Qi, C. & Xie, Y. (2013): An ant colony optimization routing algorithm for two order pickers with congestion consideration. *Computers & Industrial Engineering* 66, 77-85.

Chen, F.; Wang, H.; Xie, Y. & Qi, C. (2016): An ACO-based online routing method for multiple order pickers with congestion consideration in warehouse. *Journal of Intelligent Manufacturing* 27, 389-408.

de Koster, R.; Le-Duc, T. & Roodbergen, K. J. (2007): Design and control of warehouse order picking: A literature review. *Science Direct* 182, 481-501.

Gue, K. R.; Meller, R. D. & Skufca, J. D. (2006): The effects of pick density on order picking areas with narrow aisles. *IIE Transactions* 38, 859-868.

Hall, R. W. (1993): Distance approximations for routing manual pickers in a warehouse. *IIE Transactions* 25, 76-87.

Helsgaun, K. (2000): An effective implementation of the Lin-Kernighan traveling salesman heuristic. *European Journal of Operational Research* 126, 106-130.

Henn, S.; Koch, S.; Dörner, K. F.; Strauss, C. & Wäscher, G. (2010): Metaheuristics for the order batching problem in manual order picking systems. *BuR - Business Research* 3, 82-105.

Henn, S. & Wäscher, G. (2012): Tabu search heuristics for the order batching problem in manual order picking systems. *European Journal of Operational Research* 222, 484-494.

Henn, S. (2015): Order batching and sequencing for the minimization of the total tardiness in picker-to-part warehouses. *Flexible Services and Manufacturing* 27, 86-114.

Hirschberg, C. (2015): Variantenvielfalt oder was Automobile und Gurken gemeinsam haben. URL: http://ipl-mag.de/ipl-magazin-rubriken/scm-daten/390-variantenvielfalt

Ho, Y.-C. & Chien, S.-P. (2006): A comparison of two zone-visitation sequencing strategies in a distribution centre. *Computers & Industrial Engineering* 50, 426-439.

Hong, S.; Johnson, A. L. & Peters, B. A. (2010): Analysis of picker blocking in narrow-aisle batch picking. *Progress in Material Handling Research: Proceedings of 2010 International Material Handling Research Colloquium*, Ellis, K.P.; Gue, K.; de Koster, R.; Meller, R.; Montreuil, B.& Ogle, M. (Eds.). Charlotte: The Material Handling Institute.

Jarvis, J. M. & McDowell, E. D. (1991): Optimal product layout in an order picking warehouse. *IIE Transactions* 23, 93102.

Le-Duc, T. & de Koster, R. (2007): Travel time estimation and order batching in a 2-block warehouse. *European Journal of Operational Research* 176, 374-388.

Pan, J. C.-H. & Shih, P.-H. (2008): Evaluation of the throughput of a multiple-picker order picking system with congestion consideration. *Computers & Industrial Engineering* 55, 379-389.

Pan, J. C.-H. & Wu, M.-H. (2012): Throughput analysis for order picking system with multiple pickers and aisle congestion considerations. *Computers & Operations Research* 39, 1661-1672.

Parikh, P. J. & Meller, R. D. (2009): Estimating picker blocking in wide-aisle order picking systems. *IIE Transactions* 41, 232-246.

Parikh, P. J. & Meller, R. D. (2010): A note on worker blocking in narrow-aisle order picking systems when pick time is non-deterministic. *IIE Transactions* 42, 392-404.

Petersen, C. G. & Schmenner, R. W. (1999): An evaluation of routing and volume-based storage policies in an order picking operation. *Decision Science* 30, 481-501.

Rakrouki, M. A.; Ladhari, T. & Tkindt, V. (2012): Coupling genetic local search and recovering beamsearch algorithms for minimizing the total completion time in the single machine scheduling problem subject to release dates. *Computers & Operations Research* 39, 1257-1264.

Ratliff, H. D. & Rosenthal, A. R. (1983): Order-picking in a rectangular warehouse: A solvable case of the traveling salesman problem. *Operations Research* 31, 507-521.

Roodbergen, K. J. (2001): Layout and routing methods for warehouses. Trial: Rotterdam.

Roodbergen, K. J. & de Koster, R. (2001): Routing order pickers in a warehouse with a middle aisle. *European Journal of Operational Research* 133, 32-43.

Skufca, J. D. (2005): k workers in a circular warehouse: A random walk on a circle, without passing. *Society for Industrial and Applied Mathematics* 47, 301-314.

Statista (2016): Jährliche Anzahl der Bestellungen bei Zalando in den Jahren 2011 bis 2015. URL: https://de.statista.com/statistik/daten/studie/325862/umfrage/anzahl-aller-bestellungen-bei-zalando-pro-jahr/

Theys, C.; Bräysy, O.; Dullaert, W. & Raa, B. (2010): Using a TSP heuristic for routing order pickers in warehouses. *European Journal of Operational Research* 200, 755-763.

Tompkins, J. A.; White, J. A.; Bozer, Y. A. & Tanchoco, J. M. A. (2010): Facilities planning. 4th edition, John Wiley & Sons, New Jersey.

Van Nieuwenhuyse, I. & de Koster, R (2009): Evaluating order throughput time in 2-block warehouses with time window batching. *International Journal of Production Economics* 121, 654-664.

Yu, M.; de Koster, R. (2009): The impact of order batching and picking area zoning on order picking system performance. *European Journal of Operational Research* 198, 480-490.

Appendix: Results of the numerical experiments

Table A1: Proportion [%] of the total waiting time as part of the total processing time for A_1 and the LKH heuristic

α	(n_l, n_u)	b	m	$k=2$		$k=3$		$k=5$		$k=7$	
				$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$
3	(5,25)	1	5	3.1	3.2	6.4	7.3	13.9	17.5	21.8	33.5
3	(5,25)	1	10	1.5	2.4	3.0	5.3	6.3	12.5	9.9	23.9
3	(5,25)	2	5	1.9	2.9	3.8	6.1	8.2	15.0	12.5	28.6
3	(5,25)	2	10	0.8	1.7	1.8	3.5	3.7	7.8	5.7	13.1
3	(5,25)	3	5	1.4	2.0	2.6	4.2	6.0	9.3	9.2	15.8
3	(5,25)	3	10	0.6	1.2	1.3	2.6	2.7	5.5	4.2	9.1
3	(10,50)	1	5	3.9	4.3	8.0	9.1	16.7	22.1	26.2	40.5
3	(10,50)	1	10	1.9	2.9	3.9	6.5	8.4	14.8	13.0	28.5
3	(10,50)	2	5	2.3	3.4	4.8	7.3	9.8	17.0	15.1	33.1
3	(10,50)	2	10	1.1	1.8	2.3	3.9	4.7	8.8	7.2	14.4
3	(10,50)	3	5	1.6	2.2	3.3	4.6	6.8	10.4	10.6	16.6
3	(10,50)	3	10	0.8	1.4	1.6	2.8	3.2	6.2	5.0	10.1
10	(5,25)	1	5	4.3	5.6	9.2	12.5	18.7	32.6	28.4	50.1
10	(5,25)	1	10	1.8	4.4	4.0	9.7	8.3	25.9	12.8	44.6
10	(5,25)	2	5	2.3	4.7	4.8	10.8	9.9	28.4	15.4	47.1
10	(5,25)	2	10	1.1	2.5	2.2	5.5	4.5	12.3	6.9	20.4
10	(5,25)	3	5	1.7	3.0	3.3	6.4	7.0	14.3	10.7	22.7
10	(5,25)	3	10	0.7	1.7	1.4	3.6	3.0	8.1	4.6	13.1
10	(10,50)	1	5	5.5	7.3	11.3	16.2	22.4	39.9	33.1	56.8
10	(10,50)	1	10	2.6	5.6	5.4	12.4	11.0	32.7	16.5	51.2
10	(10,50)	2	5	2.8	6.1	5.9	13.3	12.0	35.7	18.3	53.4
10	(10,50)	2	10	1.3	3.2	2.8	6.6	5.7	15.0	8.7	23.6
10	(10,50)	3	5	1.9	3.7	4.0	7.6	8.2	16.8	12.3	26.7
10	(10,50)	3	10	0.9	2.0	1.9	4.5	3.8	9.7	5.8	15.3
20	(5,25)	1	5	5.4	8.0	11.1	18.7	22.4	43.4	33.0	59.1
20	(5,25)	1	10	2.4	6.5	5.0	15.0	10.2	38.0	15.6	54.9
20	(5,25)	2	5	2.9	7.0	5.8	16.0	11.7	40.0	17.8	56.4
20	(5,25)	2	10	1.4	3.6	2.5	7.8	5.4	17.1	8.3	27.5
20	(5,25)	3	5	2.0	4.3	4.0	8.6	8.0	19.0	12.1	29.0
20	(5,25)	3	10	0.9	2.4	1.8	4.9	3.6	10.6	5.4	17.0
20	(10,50)	1	5	6.6	9.7	13.3	21.8	26.1	47.7	36.7	62.2
20	(10,50)	1	10	3.1	7.7	6.3	18.0	12.7	44.3	19.4	59.4
20	(10,50)	2	5	3.5	8.0	6.8	19.4	14.0	44.9	20.6	60.3
20	(10,50)	2	10	1.5	4.2	3.2	9.3	6.7	19.8	10.0	31.2
20	(10,50)	3	5	2.2	4.9	4.6	10.1	9.3	21.3	14.1	33.1
20	(10,50)	3	10	1.1	2.8	2.1	5.7	4.3	12.4	6.6	20.3

Table A2: Proportion [%] of the total waiting time as part of the total processing time for A_2 and the LKH heuristic

α	(n_l, n_u)	b	m	$k=2$		$k=3$		$k=5$		$k=7$	
				$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$
3	(5,25)	1	5	2.3	0.9	4.7	2.0	9.5	4.2	13.9	6.6
3	(5,25)	1	10	1.0	0.6	1.9	1.3	3.9	2.7	5.8	4.3
3	(5,25)	2	5	1.5	0.8	2.9	1.7	5.8	3.7	8.6	5.6
3	(5,25)	2	10	0.6	1.3	1.2	2.7	2.5	5.5	3.7	8.1
3	(5,25)	3	5	1.1	1.6	2.1	3.3	4.4	6.5	6.5	10.3
3	(5,25)	3	10	0.5	0.9	1.0	1.9	1.8	3.8	2.8	5.8
3	(10,50)	1	5	3.2	1.4	6.6	2.8	12.6	5.8	19.0	9.2
3	(10,50)	1	10	1.4	0.9	2.9	1.8	5.8	3.7	8.6	5.8
3	(10,50)	2	5	1.8	1.1	3.8	2.3	7.5	4.6	11.1	7.1
3	(10,50)	2	10	0.8	1.6	1.6	3.5	3.3	6.6	5.0	9.9
3	(10,50)	3	5	1.4	1.9	2.6	4.2	5.2	8.0	7.6	11.9
3	(10,50)	3	10	0.6	1.2	1.1	2.5	2.3	4.7	3.4	7.3
10	(5,25)	1	5	3.2	1.8	6.7	3.5	13.3	7.1	19.2	10.9
10	(5,25)	1	10	1.3	1.1	2.7	2.2	5.3	5.1	8.4	7.6
10	(5,25)	2	5	1.7	1.2	3.8	2.8	7.3	5.5	11.0	8.5
10	(5,25)	2	10	0.9	2.1	1.5	4.1	3.3	8.5	4.7	12.9
10	(5,25)	3	5	1.3	2.4	2.5	5.1	5.3	10.3	7.7	15.0
10	(5,25)	3	10	0.6	1.3	1.1	2.6	2.2	5.8	3.3	8.7
10	(10,50)	1	5	4.5	2.1	9.1	4.1	17.2	8.8	25.0	13.1
10	(10,50)	1	10	1.9	1.5	3.6	2.9	7.4	5.9	11.2	9.1
10	(10,50)	2	5	2.3	1.7	4.8	3.5	9.4	6.6	13.9	10.2
10	(10,50)	2	10	1.0	2.9	2.0	5.4	3.9	11.3	6.0	16.4
10	(10,50)	3	5	1.6	3.2	3.3	6.2	6.3	12.9	9.2	18.5
10	(10,50)	3	10	0.7	1.9	1.3	3.8	2.6	7.7	4.2	11.3
20	(5,25)	1	5	4.5	2.4	8.4	4.7	16.2	9.6	23.5	14.7
20	(5,25)	1	10	1.7	1.7	3.5	3.4	7.0	7.1	10.2	11.3
20	(5,25)	2	5	2.4	2.0	4.5	3.8	8.8	8.2	13.1	12.7
20	(5,25)	2	10	1.0	3.1	2.0	5.8	3.9	11.8	6.0	18.0
20	(5,25)	3	5	1.6	3.5	3.3	6.8	6.3	13.4	9.3	19.7
20	(5,25)	3	10	0.7	1.9	1.4	3.7	3.0	7.8	4.1	11.7
20	(10,50)	1	5	5.7	2.8	10.9	5.5	20.3	10.6	28.6	16.3
20	(10,50)	1	10	2.3	1.9	4.7	4.0	9.1	8.1	13.5	12.3
20	(10,50)	2	5	3.0	2.0	5.7	4.5	10.7	9.0	15.9	13.1
20	(10,50)	2	10	1.1	3.7	2.3	7.5	4.7	15.2	7.1	21.8
20	(10,50)	3	5	1.9	4.5	3.7	8.4	7.4	16.5	10.7	23.8
20	(10,50)	3	10	0.8	2.4	1.6	5.2	3.2	10.1	4.9	15.0

Table A3: Total processing time [hours] when applying A_1 with the S-shape strategy

α	(n_l, n_u)	b	m	$k=2$		$k=3$		$k=5$		$k=7$	
				$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$
3	(5,25)	1	5	23.8	21.4	24.2	22.1	25.4	24.0	27.7	29.3
3	(5,25)	1	10	33.6	26.2	33.9	26.7	34.8	28.4	35.9	31.5
3	(5,25)	2	5	30.3	23.6	30.8	24.2	31.9	26.4	33.3	30.4
3	(5,25)	2	10	40.2	31.4	40.7	31.9	41.5	33.3	42.5	35.2
3	(5,25)	3	5	36.7	29.6	37.2	30.2	38.2	31.7	39.4	33.5
3	(5,25)	3	10	48.3	37.6	48.7	38.0	49.4	39.1	50.3	40.6
3	(10,50)	1	5	29.5	27.8	30.2	28.8	32.0	32.8	36.4	42.9
3	(10,50)	1	10	41.4	34.7	41.8	35.6	42.9	38.2	43.9	44.3
3	(10,50)	2	5	38.9	31.9	39.5	32.9	40.8	36.2	42.3	43.7
3	(10,50)	2	10	54.7	42.6	55.2	43.2	56.3	44.9	57.5	47.1
3	(10,50)	3	5	48.6	40.1	49.1	40.9	50.2	42.7	51.7	45.1
3	(10,50)	3	10	68.2	52.2	68.6	52.7	69.5	54.1	70.9	55.8
10	(5,25)	1	5	32.8	30.5	33.9	32.5	36.8	41.1	41.4	55.8
10	(5,25)	1	10	42.1	35.3	42.9	36.9	44.4	43.3	46.3	57.2
10	(5,25)	2	5	38.8	32.7	39.6	34.7	41.5	42.4	43.6	56.4
10	(5,25)	2	10	48.7	40.1	49.2	41.2	50.4	44.0	51.9	48.2
10	(5,25)	3	5	45.5	38.2	46.2	39.3	47.7	42.4	49.4	46.7
10	(5,25)	3	10	57.1	45.9	57.5	46.7	58.5	48.6	59.7	51.1
10	(10,50)	1	5	47.3	46.4	49.4	50.7	54.5	70.8	62.4	97.6
10	(10,50)	1	10	58.8	53.1	59.9	56.4	62.6	71.7	65.4	98.4
10	(10,50)	2	5	56.2	50.4	57.6	54.0	60.7	71.3	64.0	97.2
10	(10,50)	2	10	71.5	60.6	72.3	62.3	74.2	67.1	76.2	73.7
10	(10,50)	3	5	66.0	57.4	67.1	59.2	69.4	63.9	71.9	70.7
10	(10,50)	3	10	85.2	69.8	85.9	71.0	87.5	73.9	89.0	77.8
20	(5,25)	1	5	46.1	44.9	48.2	50.2	54.0	72.1	62.1	98.7
20	(5,25)	1	10	55.2	49.0	56.3	53.1	59.0	71.1	62.4	97.8
20	(5,25)	2	5	51.5	46.9	52.8	51.4	55.8	70.0	59.4	96.7
20	(5,25)	2	10	61.4	53.5	62.2	55.4	63.9	60.7	66.1	69.1
20	(5,25)	3	5	58.1	51.5	59.1	53.7	61.6	59.3	64.2	67.9
20	(5,25)	3	10	69.5	59.1	70.2	60.4	71.5	64.1	73.1	68.5
20	(10,50)	1	5	75.3	75.4	79.5	86.1	89.7	129.7	104.5	179.4
20	(10,50)	1	10	84.6	80.6	87.1	89.7	92.1	129.9	97.7	179.3
20	(10,50)	2	5	82.5	77.9	84.7	87.9	90.4	127.9	97.1	176.8
20	(10,50)	2	10	97.1	87.2	98.3	90.8	101.4	100.9	104.5	117.2
20	(10,50)	3	5	90.8	84.3	92.6	88.1	96.9	97.7	100.9	113.3
20	(10,50)	3	10	111.1	95.9	112.2	98.1	114.5	103.4	117.1	111.5

Table A4: Total processing time [hours] when applying A_2 with the S-shape strategy

α	(n_l, n_u)	b	m	$k=2$		$k=3$		$k=5$		$k=7$	
				$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$
3	(5,25)	1	5	23.7	21.1	24.1	21.3	24.9	21.7	26.0	22.3
3	(5,25)	1	10	33.5	25.8	33.8	26.0	34.5	26.4	35.2	26.9
3	(5,25)	2	5	30.2	23.2	30.5	23.4	31.3	24.0	32.0	24.7
3	(5,25)	2	10	40.1	31.3	40.4	31.6	41.0	32.3	41.7	33.1
3	(5,25)	3	5	36.6	29.3	36.9	29.6	37.6	30.2	38.2	30.8
3	(5,25)	3	10	48.3	37.4	48.4	37.6	48.9	38.2	49.4	38.8
3	(10,50)	1	5	29.4	27.3	30.0	27.6	31.3	28.3	33.6	29.0
3	(10,50)	1	10	41.3	34.2	41.6	34.5	42.3	35.0	43.1	35.5
3	(10,50)	2	5	38.9	31.4	39.3	31.8	40.4	32.8	41.4	33.9
3	(10,50)	2	10	54.7	42.5	55.0	42.9	55.8	44.0	56.8	45.1
3	(10,50)	3	5	48.5	39.8	49.0	40.2	49.8	41.1	50.9	42.0
3	(10,50)	3	10	68.1	52.0	68.4	52.3	69.1	53.0	69.9	53.7
10	(5,25)	1	5	32.6	29.7	33.4	30.2	35.3	31.3	37.5	32.6
10	(5,25)	1	10	42.1	34.5	42.7	34.9	43.9	35.8	45.2	36.9
10	(5,25)	2	5	38.6	31.8	39.2	32.4	40.5	33.5	41.8	34.9
10	(5,25)	2	10	48.6	39.9	49.0	40.5	49.9	41.8	51.0	43.4
10	(5,25)	3	5	45.4	37.7	45.8	38.2	46.9	39.3	48.0	40.6
10	(5,25)	3	10	57.0	45.6	57.3	46.1	58.0	47.1	58.7	48.2
10	(10,50)	1	5	46.8	44.6	48.3	45.5	51.7	47.5	55.8	49.5
10	(10,50)	1	10	58.6	51.4	59.4	52.1	61.3	53.5	63.1	55.0
10	(10,50)	2	5	56.0	48.7	56.9	49.6	59.2	51.8	61.6	54.1
10	(10,50)	2	10	71.4	60.2	72.1	61.4	73.6	63.8	75.2	66.8
10	(10,50)	3	5	65.8	56.5	66.7	57.2	68.6	58.9	70.5	60.8
10	(10,50)	3	10	85.1	69.3	85.8	69.9	87.0	71.4	88.2	72.9
20	(5,25)	1	5	45.7	43.0	47.2	44.0	50.6	46.4	54.5	48.8
20	(5,25)	1	10	55.1	47.2	56.1	48.0	58.2	49.9	60.4	52.2
20	(5,25)	2	5	51.3	44.9	52.2	46.0	54.3	48.1	56.4	50.9
20	(5,25)	2	10	61.3	52.8	61.9	54.0	63.4	56.6	65.0	59.6
20	(5,25)	3	5	57.9	50.6	58.7	51.4	60.4	53.5	62.1	55.8
20	(5,25)	3	10	69.4	58.7	69.9	59.5	70.9	61.5	72.0	63.4
20	(10,50)	1	5	74.1	70.8	76.9	73.3	83.3	76.9	91.1	80.7
20	(10,50)	1	10	84.1	76.7	86.0	78.1	89.5	81.4	93.2	84.9
20	(10,50)	2	5	82.0	74.0	83.9	76.2	88.2	80.2	92.6	84.8
20	(10,50)	2	10	97.0	86.1	98.1	88.3	100.6	93.1	103.3	98.7
20	(10,50)	3	5	90.6	82.5	92.0	83.9	95.1	87.0	98.1	90.6
20	(10,50)	3	10	111.0	94.8	112.0	95.9	114.0	98.6	116.2	101.4

Table A5: Total processing time [hours] when applying A_1 with the LKH heuristic

α	(n_l, n_u)	b	m	$k=2$		$k=3$		$k=5$		$k=7$	
				$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$
3	(5,25)	1	5	22.1	20.0	22.9	20.9	24.9	23.5	27.4	29.1
3	(5,25)	1	10	29.4	23.7	29.8	24.4	30.9	26.4	32.1	30.4
3	(5,25)	2	5	27.0	21.8	27.6	22.6	28.9	25.0	30.3	29.7
3	(5,25)	2	10	34.1	28.0	34.4	28.6	35.1	29.9	35.8	31.7
3	(5,25)	3	5	31.1	26.6	31.5	27.3	32.6	28.8	33.8	31.0
3	(5,25)	3	10	38.2	31.9	38.4	32.3	39.0	33.3	39.6	34.7
3	(10,50)	1	5	28.5	26.7	29.8	28.1	32.9	32.8	37.1	43.0
3	(10,50)	1	10	38.6	32.1	39.5	33.4	41.4	36.6	43.5	43.6
3	(10,50)	2	5	35.7	29.8	36.6	31.0	38.7	34.6	41.1	42.9
3	(10,50)	2	10	47.4	38.2	47.9	39.0	49.1	41.1	50.4	43.8
3	(10,50)	3	5	41.5	35.8	42.2	36.7	43.8	39.1	45.6	42.0
3	(10,50)	3	10	53.2	43.8	53.7	44.4	54.5	46.0	55.6	48.0
10	(5,25)	1	5	31.2	29.3	32.9	31.6	36.7	41.0	41.7	55.4
10	(5,25)	1	10	38.0	33.0	38.8	35.0	40.6	42.6	42.7	56.9
10	(5,25)	2	5	35.5	31.1	36.4	33.2	38.4	41.4	40.9	55.9
10	(5,25)	2	10	42.6	36.8	43.1	38.0	44.1	40.9	45.3	45.0
10	(5,25)	3	5	39.7	35.4	40.4	36.6	42.0	40.0	43.8	44.4
10	(5,25)	3	10	46.9	40.3	47.2	41.1	48.0	43.2	48.8	45.6
10	(10,50)	1	5	46.4	45.5	49.4	50.4	56.4	70.2	65.5	97.6
10	(10,50)	1	10	56.0	50.7	57.7	54.7	61.4	71.1	65.4	98.1
10	(10,50)	2	5	53.0	48.3	54.7	52.3	58.5	70.6	62.9	97.2
10	(10,50)	2	10	64.2	56.5	65.1	58.5	67.1	64.3	69.3	71.6
10	(10,50)	3	5	58.9	53.4	60.2	55.6	63.0	61.8	65.9	70.1
10	(10,50)	3	10	70.4	61.3	71.1	62.9	72.5	66.5	74.0	70.9
20	(5,25)	1	5	44.7	43.8	47.6	49.5	54.5	71.1	63.1	98.6
20	(5,25)	1	10	51.1	46.9	52.5	51.6	55.5	70.6	59.1	97.2
20	(5,25)	2	5	48.3	45.3	49.8	50.2	53.1	70.2	57.0	96.5
20	(5,25)	2	10	55.4	50.2	56.1	52.5	57.8	58.4	59.6	66.8
20	(5,25)	3	5	52.4	48.8	53.5	51.1	55.9	57.6	58.5	65.8
20	(5,25)	3	10	59.3	53.6	59.9	55.1	61.0	58.6	62.2	63.1
20	(10,50)	1	5	74.3	74.5	80.0	86.1	93.9	128.8	109.6	178.3
20	(10,50)	1	10	81.7	78.5	84.6	88.3	90.7	130.0	98.3	178.4
20	(10,50)	2	5	79.3	75.8	82.1	86.5	89.0	126.6	96.4	175.6
20	(10,50)	2	10	89.6	83.4	91.1	88.0	94.5	99.5	98.1	116.0
20	(10,50)	3	5	83.7	80.6	85.8	85.3	90.2	97.5	95.2	114.7
20	(10,50)	3	10	96.1	87.4	97.1	90.1	99.4	97.0	101.9	106.5

Table A6: Total processing time [hours] when applying A_2 with the LKH heuristic

α	(n_l, n_u)	b	m	$k=2$		$k=3$		$k=5$		$k=7$	
				$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$	$a=r$	$a=c$
3	(5,25)	1	5	21.9	19.5	22.5	19.8	23.7	20.2	24.9	20.7
3	(5,25)	1	10	29.2	23.3	29.5	23.4	30.1	23.8	30.8	24.2
3	(5,25)	2	5	26.9	21.4	27.3	21.6	28.1	22.0	29.0	22.5
3	(5,25)	2	10	34.0	27.9	34.2	28.3	34.6	29.2	35.1	30.0
3	(5,25)	3	5	31.0	26.5	31.3	27.0	32.1	27.9	32.8	29.1
3	(5,25)	3	10	38.1	31.8	38.3	32.1	38.6	32.8	39.0	33.4
3	(10,50)	1	5	28.3	25.9	29.3	26.3	31.3	27.1	33.8	28.1
3	(10,50)	1	10	38.5	31.5	39.0	31.8	40.2	32.4	41.5	33.1
3	(10,50)	2	5	35.5	29.1	36.2	29.4	37.7	30.1	39.2	31.0
3	(10,50)	2	10	47.2	38.1	47.6	38.8	48.4	40.1	49.3	41.6
3	(10,50)	3	5	41.4	35.7	41.9	36.6	43.0	38.1	44.2	39.8
3	(10,50)	3	10	53.1	43.7	53.4	44.3	54.1	45.3	54.7	46.6
10	(5,25)	1	5	30.9	28.2	32.0	28.7	34.4	29.8	36.9	31.0
10	(5,25)	1	10	37.8	31.9	38.3	32.3	39.4	33.3	40.7	34.2
10	(5,25)	2	5	35.3	30.0	36.0	30.4	37.4	31.3	38.9	32.3
10	(5,25)	2	10	42.5	36.7	42.8	37.4	43.6	39.2	44.2	41.2
10	(5,25)	3	5	39.6	35.2	40.1	36.1	41.3	38.2	42.4	40.3
10	(5,25)	3	10	46.8	40.2	47.0	40.7	47.6	42.1	48.1	43.4
10	(10,50)	1	5	45.9	43.1	48.2	44.0	52.9	46.3	58.4	48.5
10	(10,50)	1	10	55.7	48.6	56.6	49.3	59.0	50.9	61.5	52.7
10	(10,50)	2	5	52.7	46.2	54.0	47.0	56.8	48.5	59.7	50.5
10	(10,50)	2	10	63.9	56.3	64.6	57.8	65.9	61.6	67.4	65.4
10	(10,50)	3	5	58.7	53.1	59.8	54.8	61.7	59.0	63.6	63.1
10	(10,50)	3	10	70.2	61.2	70.7	62.5	71.6	65.1	72.8	67.7
20	(5,25)	1	5	44.3	41.2	46.2	42.2	50.5	44.6	55.3	47.2
20	(5,25)	1	10	50.7	44.6	51.7	45.4	53.6	47.2	55.5	49.4
20	(5,25)	2	5	48.1	43.0	49.1	43.8	51.4	45.9	54.0	48.2
20	(5,25)	2	10	55.2	49.9	55.8	51.4	56.9	54.8	58.2	59.0
20	(5,25)	3	5	52.3	48.4	53.2	50.1	54.8	53.9	56.7	58.2
20	(5,25)	3	10	59.2	53.4	59.7	54.4	60.6	56.8	61.4	59.3
20	(10,50)	1	5	73.6	69.2	77.9	71.3	87.1	75.3	97.2	80.4
20	(10,50)	1	10	81.1	73.9	83.1	75.4	87.1	78.8	91.6	82.6
20	(10,50)	2	5	78.9	71.2	81.2	73.0	85.8	76.6	91.0	80.3
20	(10,50)	2	10	89.3	82.9	90.4	86.3	92.6	94.2	95.0	102.1
20	(10,50)	3	5	83.5	80.3	85.0	83.8	88.3	91.9	91.7	100.7
20	(10,50)	3	10	95.9	87.1	96.6	89.6	98.2	94.5	100.0	99.9

Table A7: Evaluation of the truncated brach-and-bound algorithm for problem classes with 100 customer orders and S-shape strategy

a	α	b	m	$k=3$					$k=5$					$k=7$					$k=10$				
				w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2
r	10	1	5	4.1	3.0	1.9	53.9	36.8	9.1	6.3	5.4	41.2	15.0	17.1	10.4	8.3	51.1	20.1	36.1	19.8	11.0	69.5	44.4
r	10	1	10	2.2	1.7	0.7	69.4	60.3	4.8	3.6	1.6	66.9	55.2	7.6	5.3	2.6	65.8	50.9	12.9	8.0	4.4	65.9	45.0
r	10	2	5	2.6	1.9	1.1	57.6	43.6	5.7	4.2	2.6	54.0	37.3	9.0	6.6	4.7	47.2	28.3	15.2	10.6	7.5	50.4	28.5
r	10	2	10	1.5	1.4	0.6	60.0	55.8	3.4	2.8	1.4	59.2	51.0	5.4	4.4	2.4	56.5	46.5	8.8	6.9	3.7	57.4	45.8
r	20	1	5	8.6	5.9	3.5	59.0	40.6	18.8	12.4	9.8	47.7	20.4	33.5	20.2	15.7	53.2	22.3	63.4	35.1	19.8	68.7	43.5
r	20	1	10	4.7	3.6	1.4	69.8	61.2	9.7	7.1	3.2	66.9	54.5	15.3	10.8	5.3	65.3	51.1	25.3	16.4	9.1	64.1	44.6
r	20	2	5	4.7	3.9	2.1	56.0	46.7	10.4	8.1	4.9	52.9	39.6	17.0	12.5	8.0	52.6	35.6	27.8	18.9	12.7	54.4	32.8
r	20	2	10	2.6	2.4	1.1	58.3	55.6	5.6	4.9	2.4	56.6	50.3	8.8	7.6	3.9	55.4	48.4	14.6	12.0	6.7	54.0	43.8
average				3.9	3.0	1.5	60.5	50.1	8.5	6.2	3.9	55.7	40.4	14.2	9.7	6.4	55.9	37.9	25.5	15.9	9.4	60.5	41.1
c	10	1	5	7.1	1.9	0.8	88.9	59.0	27.2	3.9	2.0	92.7	49.1	54.0	5.9	3.4	93.7	42.6	93.5	9.0	5.6	94.0	37.5
c	10	1	10	5.7	1.4	0.6	89.7	59.0	21.0	2.8	1.5	93.1	47.2	47.7	4.3	2.4	94.9	44.3	87.7	7.0	4.1	95.3	40.8
c	10	2	5	6.4	2.0	1.0	84.4	50.1	23.7	4.1	2.4	89.8	41.6	49.6	6.5	4.3	91.4	34.8	88.9	10.7	6.7	92.4	37.0
c	10	2	10	3.0	2.1	1.1	63.7	48.5	7.8	4.6	3.0	61.7	34.2	14.5	7.5	5.0	65.6	33.8	32.6	12.7	8.0	75.6	37.0
c	20	1	5	17.4	4.6	1.7	90.0	61.8	61.0	8.2	4.0	93.4	50.9	110.7	12.0	7.2	93.5	40.5	183.4	18.3	11.5	93.7	37.1
c	20	1	10	14.6	3.0	1.3	90.9	55.5	54.7	6.2	2.9	94.6	52.8	104.1	9.7	5.3	94.9	45.3	178.3	15.1	8.8	95.0	41.6
c	20	2	5	15.8	4.1	1.9	87.7	52.7	55.8	8.2	4.8	91.3	40.7	104.7	12.7	7.9	92.5	38.0	176.8	20.1	12.6	92.9	37.1
c	20	2	10	6.4	3.9	2.1	66.5	45.6	16.5	8.7	5.7	65.7	35.2	32.8	14.3	10.0	69.4	29.8	70.2	25.0	14.4	79.4	42.4
average				9.5	2.9	1.3	82.7	54.0	33.5	5.8	3.3	85.3	44.0	64.8	9.1	5.7	87.0	38.6	114.0	14.7	9.0	89.8	38.8

Table A8: Evaluation of the truncated brach-and-bound algorithm for problem classes with 100 customer orders and LKH heuristic

a	α	b	m	$k=3$					$k=5$					$k=7$					$k=10$				
				w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2	w_1	w_2	w_B	imp_1	imp_2
r	10	1	5	5.6	4.4	2.4	57.5	45.4	12.6	9.1	5.6	55.6	38.5	21.7	14.6	8.2	62.2	43.7	36.3	22.7	11.6	68.1	48.9
r	10	1	10	3.1	2.1	1.1	64.1	45.9	6.8	4.4	2.5	62.8	42.6	10.8	6.9	3.9	63.6	43.1	18.0	10.6	6.4	64.4	39.8
r	10	2	5	3.2	2.6	1.3	60.6	50.2	7.0	5.3	2.9	58.9	45.6	11.5	8.3	4.7	59.4	43.7	18.8	12.8	7.2	61.4	43.5
r	10	2	10	1.8	1.3	0.7	62.4	48.5	3.9	2.6	1.4	63.3	45.6	6.0	4.1	2.2	63.3	45.4	9.6	6.1	3.4	64.3	43.9
r	20	1	5	10.6	8.5	4.5	57.2	46.4	24.5	17.8	10.5	57.3	41.4	40.2	27.9	16.2	59.7	41.9	66.3	43.4	21.7	67.3	50.0
r	20	1	10	5.3	3.9	2.1	61.1	46.3	11.5	8.0	4.5	61.1	43.8	19.0	12.4	7.3	61.7	41.0	31.4	19.2	11.1	64.6	42.1
r	20	2	5	5.6	4.6	2.4	57.6	48.9	12.5	9.2	5.2	58.4	43.8	19.9	14.6	8.4	57.6	42.4	32.5	22.6	12.8	60.5	43.2
r	20	2	10	2.9	2.1	1.1	62.5	48.9	6.3	4.4	2.3	62.7	46.1	9.8	6.7	3.6	63.1	46.0	15.3	10.2	5.8	62.2	43.3
average				4.8	3.7	1.9	60.4	47.6	10.6	7.6	4.4	60.0	43.4	17.4	11.9	6.8	61.3	43.4	28.5	18.5	10.0	64.1	44.3
c	10	1	5	8.2	1.8	1.0	88.0	46.3	28.0	4.1	2.4	91.4	40.5	55.4	6.3	4.1	92.6	35.3	94.1	10.7	6.4	93.2	40.4
c	10	1	10	6.8	1.5	0.7	89.1	49.3	23.2	3.0	1.9	91.8	36.9	50.2	4.8	3.1	93.9	36.4	89.8	8.0	5.1	94.3	36.1
c	10	2	5	7.0	1.6	0.8	87.9	47.7	25.2	3.2	1.9	92.5	40.3	51.9	5.2	3.4	93.4	33.9	90.7	8.5	5.4	94.0	36.1
c	10	2	10	3.9	3.1	1.6	58.9	49.0	9.6	7.0	4.0	58.3	42.3	16.9	10.7	6.5	61.3	38.7	33.1	18.2	9.4	71.7	48.7
c	20	1	5	18.8	3.9	2.1	88.7	46.2	61.5	8.0	5.0	91.9	37.5	111.0	13.1	8.3	92.5	36.5	182.3	21.1	12.6	93.1	40.5
c	20	1	10	15.9	3.0	1.6	90.1	47.8	57.6	6.4	3.8	93.3	39.8	105.9	10.1	6.5	93.8	35.7	179.4	16.4	10.2	94.3	38.0
c	20	2	5	16.8	3.3	1.7	89.9	48.2	56.9	6.9	3.9	93.1	42.8	105.9	10.5	6.6	93.8	37.4	177.0	17.1	10.8	93.9	36.5
c	20	2	10	8.2	6.5	3.3	60.0	49.4	19.7	14.2	8.5	56.9	40.3	36.1	22.2	13.1	63.8	41.0	67.9	36.1	17.5	74.1	51.4
average				10.7	3.1	1.6	81.6	48.0	35.2	6.6	3.9	83.6	40.1	66.7	10.4	6.5	85.6	36.9	114.3	17.0	9.7	88.6	41.0

Part VI:

Order Batching and Picker Routing for the Minimization of the Total Travel Distance

CEJOR
DOI 10.1007/s10100-017-0467-x

ORIGINAL PAPER

Order Batching and Picker Routing in manual order picking systems: the benefits of integrated routing

A. Scholz[1] · G. Wäscher[1,2]

Abstract Order Batching and Picker Routing Problems arise in warehouses when items specified by customer orders have to be retrieved from their storage locations. The Order Batching Problem includes the grouping of a given set of customer orders into feasible picking orders such that the total length of all picker tours is minimized. In order to calculate the length of a picker tour, the sequence has to be determined according to which the items contained in the picking order will be picked. This problem is known as the Picker Routing Problem. Although quite sophisticated heuristics and even efficient exact solution approaches exist to the Picker Routing Problem in warehouse with up to two blocks, the routing problem does not get much attention when dealing with the Order Batching Problem. Instead, the order pickers are assumed to follow a certain, simple routing strategy when making their ways through the warehouse. The advantage of this approach can be seen in the fact that—in particular for single-block warehouse layouts—the corresponding picker tours are very straightforward and can be memorized easily by the order pickers. This advantage diminishes, however, when more complex, multi-block layouts have to be dealt with. Furthermore, in such case, the approach may result in picker tours which are far from optimal. For multi-block layouts, we integrate different routing algorithms into an iterated local search approach for the batching in order to demonstrate what the benefits are from solving the Order Batching and the Picker Routing Problem in a more integrated way. By means of numerical experiments it is shown that paying more attention to the

✉ A. Scholz
andre.scholz@ovgu.de

1 Faculty of Economics and Management, Otto-von-Guericke University Magdeburg, Postbox 4120, 39016 Magdeburg, Germany

2 School of Mechanical, Electronic and Control Engineering, Beijing Jiaotong University, 100044 Beijing, China

Published online: 31 January 2017

Picker Routing Problem results in a substantial improvement of the solution quality without increasing computing times.

Keywords Order Picking · Order Batching · Picker Routing · Iterated local search

1 Introduction

Order picking is a warehouse function dealing with the retrieval of items from their storage locations in order to satisfy a given demand specified by customer orders (Petersen and Schmenner 1999; Wäscher 2004). It occurs because incoming articles are received and stored in large volume unit loads while customer orders comprise small volumes (less-than-unit-loads) of different articles. In spite of various attempts to automate the picking process, approximately 80% of all order picking systems in Western Europe are manual ones (de Koster et al. 2007). Among such systems, picker-to-parts systems can be looked upon as the most important ones, where order pickers walk (or ride) through the warehouse and collect the requested items from their storage locations (Wäscher 2004). Due to a large-scale involvement of human operators, order picking includes the most cost-intensive warehouse operations. According to the literature, between 50% (Frazelle 2002) and 65% (Coyle et al. 1996) of the total warehouse operating costs can be attributed to order picking.

On the operational level, the Order Batching Problem and the Picker Routing Problem represent the key planning problems for operating distribution warehouses efficiently (de Koster et al. 1999a). In the Order Batching Problem, a set of (indivisible) customer orders is given, each of which requiring certain items to be collected from known storage locations within the warehouse. These customer orders have to be grouped into picking orders (batches) in such a way that the total length of all picker tours necessary to collect all items is minimized. In order to determine the length of a picker tour, it has to be decided in which sequence the storage locations of the respective items have to be visited. This gives rise to the so-called Picker Routing Problem. In the Picker Routing Problem, a picking order is given, and one has to determine a tour of minimum length that allows for collecting all items included in the picking order.

From this point of view, solving the Picker Routing Problem is dependent on the solution of the Order Batching Problem. However, in order to solve the Order Batching Problem, certain assumptions have to be made with respect to the routing schemes according to which the order pickers are guided on their tours through the warehouse. Therefore, efficient operation of order picking systems requires careful analysis of the two problems and their interdependencies and an integrated solution of both problems.

As individual problems, both the Order Batching Problem and the Picker Routing Problem have been studied quite extensively (de Koster et al. 2007). However, despite the fact that sophisticated exact and heuristic algorithms exist for its solution, the Picker Routing Problem does not receive much attention when the two problems are to be solved in an integrated way. It is usually argued that, in practice, order pickers follow a simple routing strategy when making their way through the warehouse. Based on this routing strategy, the customer orders are grouped into picking orders and the total tour length is determined. In order to justify the application of such routing strategies, it is argued that the provided solutions (tours) appear to be “more straightforward”; they can probably be memorized more easily than optimal ones and, therefore, get more easily accepted by the order pickers (de Koster et al. 1999a). This reasoning is questionable, though. First, the solution quality of such routing strategies is strongly dependent on the problem data (e.g. number of picking aisles of the warehouse, capacity of the picking device), and it is not uncommon that they provide tours which are far from being optimal. Second, for more complex layouts (e.g. layouts with two or more blocks), even these routing strategies may result in tours which are not straightforward at all but equally complex as optimal tours (see Roodbergen and de Koster (2001b) for examples). In other words, the core argument in favor of heuristic routing strategies, the simple structure of the provided tours, is no longer valid. Third, also the quality of the solutions generated by means of simple routing strategies tends to deteriorate when more complex warehouse layouts are considered (Roodbergen 2001).

We conclude that, for efficient order picking in more complex, multi-block layouts, more attention should be paid to the arising Picker Routing Problems when dealing with the Order Batching Problem. Hence, we propose an approach which allows for solving the Order Batching Problem and Picker Routing Problem in a more integrated way, and we will demonstrate what the benefits are from such an approach. In order to keep the exposition simple, the approach will be described and exemplified for two-block layouts, only. Nevertheless, it will become clear that the approach can be extended to block layouts of higher dimensions and that the observed benefits hold or even increase for such layouts.

We further note that, on the one hand, more sophisticated routing algorithms will result in shorter tours and, consequently, in better solutions. On the other hand, however, the integration of complex routing algorithms may result in longer, probably unacceptable computing times. We will, therefore, also study under which conditions exact routing algorithms can be applied, and we will suggest heuristic modifications of an exact routing algorithm for situations in which this is not possible.

The remainder of this paper is organized as follows: In Sect. 2, we give a precise statement of the problem under discussion, which we name the Joint Order Batching and Picker Routing Problem. Section 3 comprises a literature review regarding the routing and the batching problem. For the Joint Order Batching and Picker Routing Problem, a corresponding mathematical model formulation based on Gademann and van de Velde (2005) is presented in Sect. 4. Section 5 contains solution approaches to the Picker Routing Problem, namely the exact algorithm by Roodbergen and de Koster (2001a), the S-shape, largest gap, aisle-by-aisle and combined$^+$ heuristic. Furthermore, a heuristic solution approach derived from the exact algorithm is proposed in order to close the gap between the complex exact algorithm and the very simple routing strategies. In Sect. 6, an iterated local search algorithm for the Order Batching Problem is introduced which allows for integrating different routing algorithms. Section 7 is devoted to the numerical experiments which have been carried for evaluating the impact of the routing algorithms used in the iterated local search approach on the solution quality. We explain how the test problem instances were generated, and we report and discuss the results from the experiments. The paper concludes with an outlook on potential areas of future research.

2 Problem description

In the following, we consider a distribution warehouse with a manual, low-level picker-to-part order picking system from which a given set of items has to be retrieved. In a manual order picking system, human operators perform the necessary tasks. In picker-to-parts systems the order pickers walk (or ride) through the picking area, visit the storage locations of the respective articles (pick locations), and remove (pick) the required number of article units (items). In low-level picker-to-parts systems the items have to be removed from pallets or bins placed on the floor or from low-level racks which are directly accessible to the order pickers (Henn et al. 2012). The picking area possesses a block layout, i.e. it consists of a certain number of straight parallel picking aisles of equal length and width. The storage locations are of identical size and arranged on both sides of the picking aisles. By means of cross aisles, the order pickers are enabled to enter and exit a picking aisle. The part of the warehouse located between two adjacent cross aisles is called a block and the corresponding part of the picking aisle is denoted as a subaisle, i.e. each picking aisle is composed of q subaisles, where q denotes the number of blocks. The order pickers enter the picking area at the depot. This is also the place where they return to in order to deposit the picked items. An example of a block layout with two blocks is depicted in Fig. 1. Warehouses with two blocks contain three cross aisles, namely the front, middle and rear cross aisle, where the front cross aisle presents the cross aisle nearest to the depot.

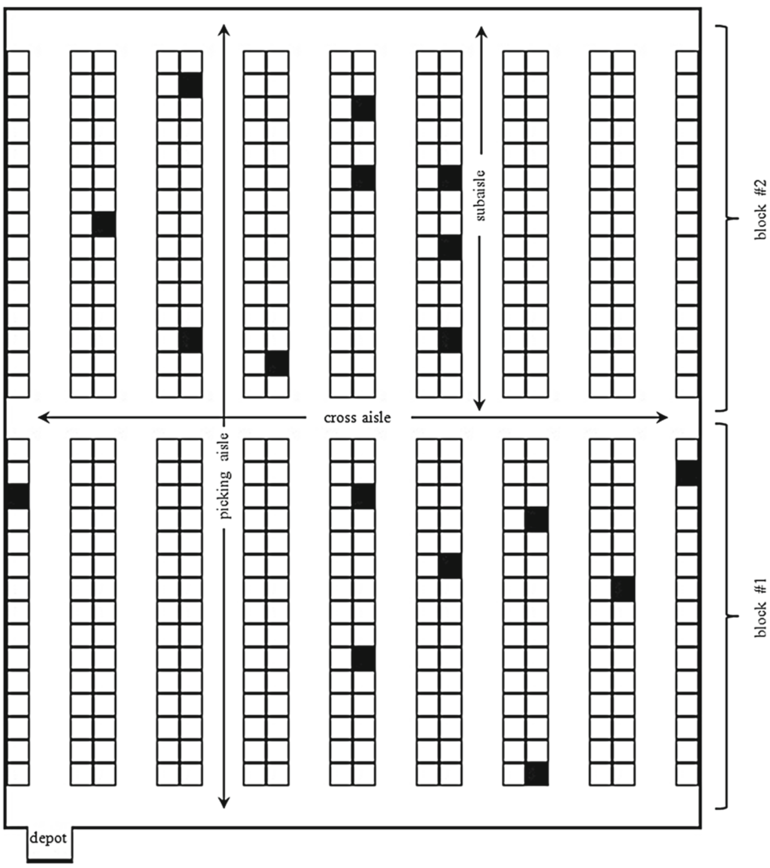


Fig. 1 Two-block layout

When moving through the warehouse, the order pickers use a picking device (e.g. a cart or a roll cage) on which they can place several items. Thus, they may visit several pick locations before they return to the depot. In other words: the requested items are picked on tours through the warehouse. The number of stops on each tour is limited by the available space of the picking device on the one hand and by the capacity requirements of the items to be picked on the other hand. The information about the items to be retrieved from the warehouse is comprised in a set of customer orders. Each of these orders is related to a particular (external or internal) customer and consists of a set of order lines, each one identifying a particular article and the corresponding quantity. All items requested in the customer orders should be retrieved in such a way that the total length of the necessary tours (total tour length) is minimized.

On their tours through the warehouse, order pickers are guided by pick lists. A pick list contains the order lines which should be processed together (picking order) and the storage locations of the respective articles. Furthermore, the order lines are already sorted into the sequence according to which the order picker is meant to visit the pick locations. A picking order may be composed of a combination of multiple customer orders. Splitting of customer orders, however, is not permitted since it would result in an additional, unacceptable consolidation effort. In this case a picking order is addressed as a batch.

Let a (non-empty) set of customer orders be given, each of which requiring certain items with known storage locations to be retrieved from a distribution warehouse. Then, in order to minimize the total length of the tours necessary to collect all requested items, the following two questions have to be answered:

- How should the given set of customer orders be grouped (batched) into picking orders? (Order Batching Problem)
- For each picking order, in which sequence should the storage locations of the articles be visited which are included in the respective order? (Picker Routing Problem)

Obviously, both problems are closely interconnected and, therefore, should be solved in an integrated way, giving rise to the Joint Order Batching and Picker Routing Problem (JOBPRP). Before we provide a model formulation to the JOBPRP, we will review the state-of-the-art related to the individual problems.

3 Literature review

3.1 Picker routing problem

Given a set of items to be collected and the respective locations where they are stored in the warehouse, a sequence must be determined according to which these locations should be visited such that the length of the corresponding tour is minimized. This so-called Picker Routing Problem (PRP) represents a special case of the classic Traveling Salesman Problem (TSP), i.e. solution approaches to the TSP can be applied in order to deal with the PRP. Due to the limited capacity of the picking device, the number of pick locations to be visited in a tour is quite small in practical applications and, therefore, the PRP can even be solved to optimality within a short amount of computing time by using an exact approach to the TSP. However, dependent on the layout of the picking area, efficient problem-specific solution approaches to the PRP exist, which are able to solve any practical-sized problem instance within fractions of a second and clearly outperform any exact TSP algorithm in terms of computing time (Ratliff and Rosenthal 1983; Roodbergen and de Koster 2001b).

For single-block layouts, Ratliff and Rosenthal (1983) presented an exact algorithm with complexity $O(n+m)$ where n stands for the number of pick locations and m for the number of picking aisles. In practice, though, instead of proceeding according to optimal tours, which are often looked upon as complicated and difficult to memorize, order pickers seem to prefer tours based on simple routing strategies. The application of such routing strategies, e.g. the S-shape, the return, the midpoint and the largest gap strategy, can be considered as a heuristic approach to the PRP. As for the S-shape heuristic (Goetschalckx and Ratliff 1988), the order picker entirely traverses every picking aisle containing at least one item to be picked. When following the return heuristic, the order picker enters and leaves every picking aisle to be visited from the front cross aisle. For the midpoint heuristic, the warehouse is divided equally into two areas, the front section and the rear section. All items located in the front section are accessed from the front cross aisle, while items in the rear section are reached from the rear cross aisle. For the largest gap heuristic, in each aisle to be visited the two warehouse sections are determined by the largest distance between two pick locations or between a pick location and the adjacent cross aisle. Hall (1993) noticed that the largest gap heuristic outperforms the midpoint heuristic. Heuristics based on more sophisticated routing strategies are the composite heuristic (Petersen 1997) and the aisle-by-aisle heuristic (Vaughan and Petersen 1999) which combine elements of the S-shape and the return heuristic. These two heuristics have in common that, for each picking aisle where an item has to be picked, it has to be decided whether the order picker entirely traverses the picking aisle or, alternatively, enters and leaves it via the same cross aisle. In the aisle-by-aisle heuristic, this decision is derived by means of dynamic programming. In the composite heuristic, a picking aisle is traversed if more than half of the picking aisle has to be entered to pick all items in this aisle.

All above-mentioned approaches to the PRP have been designed initially for single-block layouts. Roodbergen and de Koster (2001b) modified the Ratliff-Rosenthal Algorithm for the PRP in two-block layouts. For two-block and more complex layouts, the authors also introduced extensions of the S-shape and the largest gap heuristic. Furthermore, they proposed the so-called combined heuristic, which includes elements of the aisle-by-aisle heuristic. Theys et al. (2010) applied different TSP heuristics to various classes of PRP instances related to warehouses with two and more blocks. They report significant (average) savings in the total tour length (up to 48% in comparison to tours constructed by the S-shape strategy) when using the Lin–Kernighan–Helsgaun Heuristic (Helsgaun 2000).

3.2 Order Batching Problem

The Order Batching Problem (OBP) can be stated as follows (Wäscher 2004; Henn and Wäscher 2012): Given the article storage locations, the routing strategy to be used, and the capacity of the picking device, how can the set of customer orders be grouped into picking orders such that the sum of the total lengths of all tours required to collect the requested items is minimized?

The OBP as described above is known to be NP-hard (in the strong sense) if the number of orders per batch is greater than two (Gademann and van de Velde 2005). Hence, it is not surprising that only a few exact solution approaches have been proposed so far for this problem. Gademann and van de Velde (2005) presented a branch-and-price algorithm with column generation that was able to solve problem instances with up to 32 customer orders to optimality. Bozer and Kile (2008) proposed a mixed-integer programming approach that provided optimal solutions for instances with up to 25 orders within a few minutes. However, their approach is limited to S-shape routing and, thus, not suitable for the JOBPRP.

Apart from these exact approaches, a large variety of heuristic solution approaches exists. (For a detailed review, we refer to de Koster et al. (2007) and Henn et al. (2012).) The most prominent ones are priority rule-based algorithms, seed algorithms, savings algorithms and metaheuristics. In priority rule-based algorithms, priorities are assigned to customer orders at first; then, in the sequence given by the priorities, the customer orders are successively allocated to batches as long as the capacity of the picking device is not exceeded (Gibson and Sharp 1992). Seed algorithms have been introduced by Elsayed (1981). Batches are generated sequentially in such algorithms. The procedure for each batch is as follows: In a first step, a seed (or initial) order is chosen from those orders not yet added to a batch. Then, in a second step, not yet selected customer orders are added to the seed order until no further order can be added without violating the capacity constraint of the picking device. Savings algorithms are based on the Clarke-and-Wright Algorithm for the Vehicle Routing Problem (Clarke and Wright 1964) which has been adapted for the OBP. The initial version can be described as follows: At the beginning, savings (in terms of reductions of the total travel distance) are determined which can be obtained by collecting items of two customer orders on a single (large) tour instead of collecting them on two separate tours. Then, in a non-ascending order of the savings, the pairs of customer orders are assigned to batches in a way in which one tries to include both orders in the same batch. The algorithm terminates when each order has been assigned to a batch or when all pairs of orders have been considered. A straightforward improvement of the algorithm consists of recalculating the savings each time a customer order is added to a batch (Elsayed and Unal 1989). By means of numerical studies, de Koster et al. (1999b) have shown that, among these constructive algorithms, either seed or savings algorithms provide the best solutions.

Several metaheuristics have also been proposed for the OBP. Tsai et al. (2008) presented a genetic algorithm which is based on the assumption that splitting customer orders is allowed and, therefore, deals with a problem different to the one introduced above. Gademann and van de Velde (2005) suggested an iterated improvement algorithm in which, starting from an initial solution constructed by means of the first-come-first-served rule, improved solutions are obtained by so-called swap moves, i.e. by interchanging two customer orders from two different batches. They were able to solve instances with up to 30 customer orders within a few seconds. Albareda-Sambola et al. (2009) developed a variable neighborhood search algorithm for the OBP in which three different kinds of neighborhood structures are considered. This approach was applied to four different problem classes including up to 250 orders, and the authors reported computing times of less than 1 min. Henn et al. (2010) proposed an iterated local search algorithm and a rank-based ant system. Comprehensive numerical studies have shown that these approaches lead to very good solutions but also require long computing times (up to 20 min) for large instances (up to 60 customer orders). Henn and Wäscher (2012) described a tabu search algorithm and an attribute-based hill climber approach to the OBP which outperform the iterated local search algorithm in terms of solution quality and require computing times of up to 2 min (classic tabu search) and up to 10 min (attribute-based hill climber) for problem instances with up to 100 customer orders.

The above mentioned approaches integrate very simple routing policies (mainly S-shape and largest gap strategies) into the respective batching heuristic. So far, only few articles exist taking other routing methods into account. Kulak et al. (2012) dealt with the OBP in a single- and a two-block block layout where the depot is located at the middle cross aisle instead of the front cross aisle. The authors introduced a tabu search algorithm for the OBP and integrated two TSP heuristics in order to solve the arising PRPs. As TSP heuristics, they used a nearest neighbor and a savings heuristic which also represent rather simple routing algorithms. Grosse et al. (2014) proposed a simulated annealing algorithm for the batching problem and combined it with four different routing heuristics. These routing heuristics were also used to create initial batches. However, when creating the initial batches by means of a routing algorithm, it is not taken into consideration to which customer order a requested item is assigned implying that items included in the same customer order may be contained in different batches. This would not represent a feasible solution to the variant of the JOBPRP considered in this paper since splitting of customer orders is not allowed.

4 Model formulation

In order to formulate a model and develop a solution approach to the JOBPRP, we assume that each article has been assigned to exactly one storage location as also done by Kulak et al. (2012) and Grosse et al. (2014). A straightforward way to formulate a model for the JOBPRP consists of considering—either explicitly or implicitly—all feasible batches and to choose some of them such that each customer order is contained in exactly one batch and the total tour length is minimized. A model based on this approach has been introduced by Gademann and van de Velde (2005) and includes the following parameters:

The set J contains all customer orders and the set F all feasible batches. A batch is called feasible if it does not violate the capacity constraint resulting from the picking device. The constants a_{fj} indicate whether a customer order $j \in J$ is included in batch $f \in F (a_{fj} = 1)$ or not $(a_{fj} = 0)$. Furthermore, the constants d_f represent the minimum length of an order picking tour which includes all items of customer orders contained in batch $f \in F$. In order to calculate the constants d_f, for each feasible batch $f \in F$, the corresponding PRP has to be solved. Finally, the variables

x_f indicate whether batch $f \in F$ is chosen ($x_f = 1$) or not ($x_f = 0$). Now the mathematical model for the JOBPRP can be formulated as follows:

$$\min \sum_{f \in F} d_f \cdot x_f \tag{1}$$

$$\sum_{f \in F} a_{fj} \cdot x_f = 1, \qquad \forall j \in J; \tag{2}$$

$$x_f \in \{0, 1\}, \qquad \forall f \in F. \tag{3}$$

Constraints (2) and (3) ensure that a set of batches is chosen in such a way that each customer order is contained in exactly one of these batches. The objective function represents the total tour length caused by the chosen batches.

In order to solve the above-presented mathematical model, it is necessary to consider (at least implicitly) all feasible batches. Furthermore, for each feasible batch, the minimum length of the corresponding order picking tour has to be calculated. If a two-block layout of the warehouse is assumed, the PRP can be solved efficiently (Roodbergen and de Koster 2001a). However, the number of feasible batches $|F|$ grows exponentially with the number of customer orders $|J|$. Hence, only very small problems can be solved by applying the model formulation presented above.

For this reason, other approaches to the JOBPRP have to be considered. In order to deal with the JOBPRP, we introduce an iterated local search approach to the OBP developed by Henn et al. (2010) and integrate different routing algorithms. The integration of more sophisticated routing approaches results in shorter tours and, therefore, leads to better solutions to the JOBPRP. However, simple routing strategies require much less computational effort and allow the iterated local search approach for investigating a larger part of the solution space within the same amount of computing time. Due to this reason, it is not obvious whether it is better to put most effort in dealing with the OBP, as it is done in the literature so far, or to pay more attention to the solution of the arising PRPs. In the following section, we present different solution approaches to the PRP, including the exact algorithm for the PRP in warehouses with two blocks provided by Roodbergen and de Koster (2001a) as well as some heuristic routing strategies. Furthermore, we heuristically modify the exact algorithm in order to obtain an approach requiring less computational effort than the exact approach, while providing better solutions than the routing strategies.

5 Solution approaches to the PRP in two-block layouts

5.1 Exact solution approach

The PRP in warehouses with a two-block layout can be solved efficiently (Roodbergen and de Koster 2001a). The corresponding exact solution approach will be presented briefly in the following. For a more detailed presentation of this algorithm we refer to Roodbergen and de Koster (2001a).

The solution approach consists of a dynamic programming procedure. The tour is constructed by starting with an empty tour and consecutively adding subaisles to the

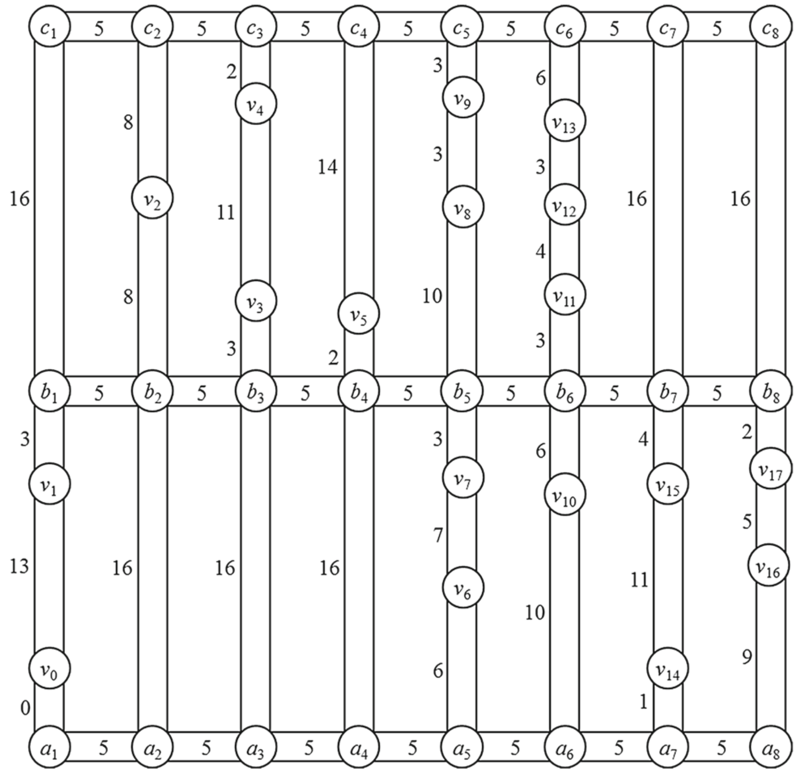


Fig. 2 Graph representing the Picker Routing Problem of Fig. 1

tour until all subaisles to be visited are included. The problem is represented by a graph $G = (V, E)$ with the set of edges E and the set of vertices $V = \bigcup_{j=1}^{m} \{a_j, b_j, c_j\} \cup \{v_0, \ldots, v_n\}$. v_0 represents the location of the depot and the vertex v_i represents the storage location of item $i \in \{1, \ldots, n\}$. Vertices a_j, b_j and c_j correspond to the points where picking aisle $j \in \{1, \ldots, m\}$ can be accessed, namely via the rear cross aisle (a_j), the middle cross aisle (b_j) and the front cross aisle (c_j), respectively. Each pair of adjacent positions (either two pick locations or a pick location and the adjacent cross aisle) is connected by two parallel edges. The corresponding edge weight is equal to the distance between these two positions. Figure 2 depicts the graph related to the PRP introduced in Fig. 1.

From the graph G, edges have to be chosen such that they correspond to a feasible tour and the sum of the edge weights is minimized. A tour is called feasible if each vertex v_i ($i \in \{0, \ldots, n\}$) is included in the tour, each vertex has an even or a zero degree and, excluding vertices with zero degree, the subgraph is connected. Thus, an order picking tour can be perceived as a subgraph of G. Since an order picking tour is constructed by consecutively extending a subgraph of G, the following definitions are needed. Let L_j^- be a subgraph of G consisting of vertices a_j, b_j and c_j together with all vertices and (some) edges corresponding to the left of picking aisle j. Furthermore, L_j^{+1} denotes a subgraph of G which additionally contains all vertices and (some) edges corresponding to subaisle j of block #1. L_j^{+2} denotes a subgraph that contains all vertices and (some) edges corresponding to the left of picking aisle $j + 1$ except edges connecting aisle j and aisle $j + 1$. The denotation L_j is used to indicate that a result holds for $L_j = L_j^-$, $L_j = L_j^{+1}$ or $L_j = L_j^{+2}$.

A subgraph T_j of L_j is called a L_j partial tour subgraph (L_j-PTS), if at least one subgraph of G (called completion) exists consisting of vertices and edges not included in L_j, such that the union of T_j and the completion leads to a graph representing an order picking tour. Hence, we can find an optimal tour by considering a picking aisle j, constructing all L_j-PTSs (either L_j^--, L_j^{+1}- or L_j^{+2}-PTSs) and identifying the best completion for each PTS. In general, it is quite difficult to find the best completion, i.e. the completion with a minimum sum of edge weights, but we could consider L_m^{+2}-PTSs, where a graph with an empty set of edges would represent the best completion. However, the consideration of all PTSs would lead to an unacceptable computational effort. Fortunately, not all PTSs have to be taken into consideration since out of two L_j-PTSs with the same set of completions only the shorter subgraph, i.e. the subgraph with the smaller sum of edge weights, may lead to an optimal tour. L_j-PTSs having an equal set of completions are called equivalent. Roodbergen and de Koster (2001a) have shown that 25 different equivalent classes exist which have to be taken into consideration. This theorem leads to the following solution approach:

The algorithm starts by constructing a L_1^-- PTS, i.e. a subgraph $T = (\tilde{V}, \tilde{E})$ with $\tilde{V} = \{a_1, b_1, c_1\}$ and $\tilde{E} = \{\emptyset\}$. Then, the first subaisle of block #1 is considered and edges are added to T in such a way that each vertex in this subaisle is incident to at least two edges. Therefore, for the construction of an optimal tour, only six different edge combinations have to be considered (Roodbergen and de Koster 2001a). These edge combinations are depicted in Fig. 3a. Configurations (1) and (6) indicate that a subaisle is traversed once or twice, respectively. The edge configurations (3) and (4) correspond to return strategies, i.e. the requested items are picked by entering and leaving a subaisle via the same cross aisle, and configuration (5) follows a largest gap policy. Configuration (2) can only be used if a subaisle does not contain any requested items. Otherwise, the inclusion of this configuration would not lead to a PTS. Adding each of configurations (1) to (6) to a L_1^--PTS results in different L_1^{+1}-PTSs. Configurations (1) to (6) are then added to each of these PTSs representing the best PTS of its corresponding equivalent class in order to receive L_1^{+2}-PTSs. Again, we determine the PTSs representing the subgraph with a minimum sum of edge weights in its equivalent class. These subgraphs contain all vertices corresponding to pick locations in the first picking aisle.

In order to obtain L_2^--PTSs, edge configurations representing the order picker changing over from one picking aisle to another have to be added to these L_1^{+2}-PTSs. By definition, each vertex included in an order picking tour must have an even degree. Therefore, picking aisles have to be connected by an even number of edges. Furthermore, an optimal tour contains at most two edges between two vertices. This leads to 14 different edge configurations that can be used to change over from one picking aisle to another (Roodbergen and de Koster 2001b). Some of these configurations are depicted in Fig. 3b.

For each equivalent class, the best L_2^--PTS is determined and configurations (1)–(6) are added to these subgraphs which results in L_2^{+1}-PTSs. Following this procedure, we finally obtain L_m^{+2}-PTSs. Now we get an optimal tour by determining the PTS which results in a minimum sum of edge weights and corresponds to a feasible tour.

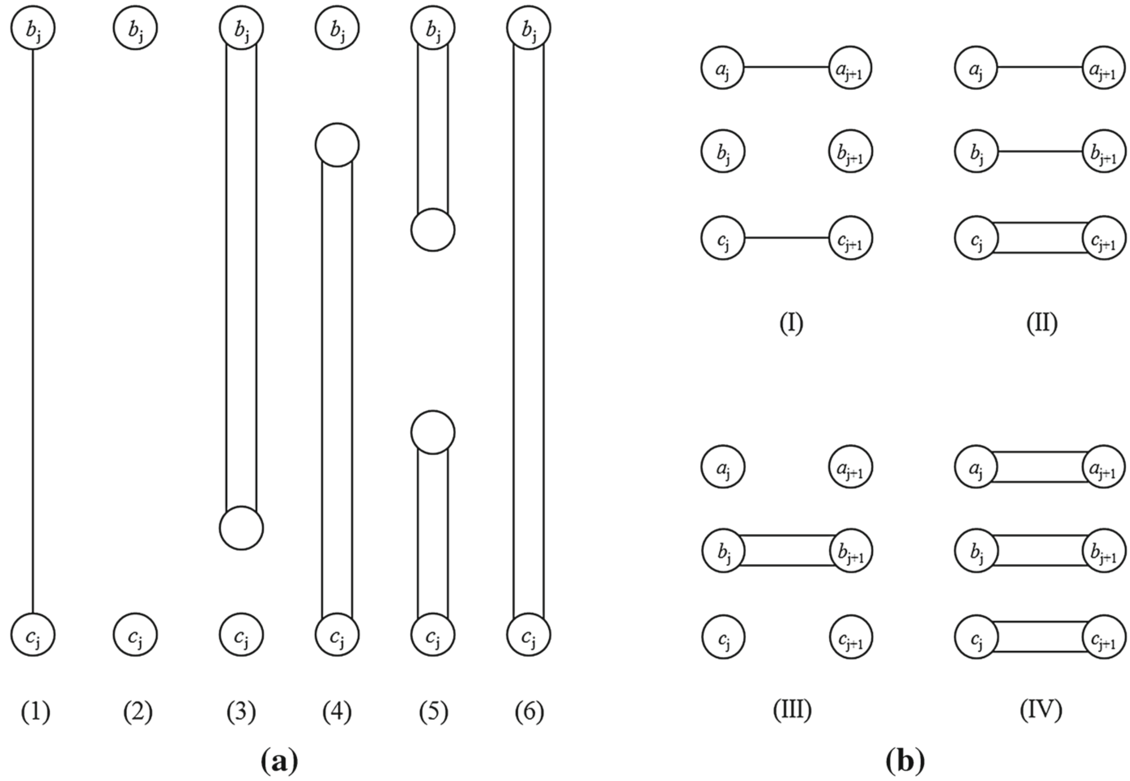


Fig. 3 **a** All possible edge configurations for traversing a subaisle. **b** A selection of configurations for changing between aisles

A pseudo-code of the algorithm is given in the appendix (available at http://www.mansci.ovgu.de/mansci/en/Research/Materials/2015+_+I_-p-472.html).

An optimal tour provided by this solution approach is depicted in Fig. 4. The time-complexity function of this exact solution approach is linear in the number of picking aisles m and the number of pick locations n (Roodbergen and de Koster 2001a) and, therefore, application of the exact algorithm to a single instance of the PRP will not result in unacceptable computing times. Even large instances can be solved within fractions of a second. However, when applied within a complex heuristic algorithm for the JOBPRP, where many PRPs have to be solved repeatedly, the computing time needed by the exact algorithm can become a critical issue. For example, Henn et al. (2010) have reported computing times up to 20 min for solving an OBP with 60 orders by applying a rank-based ant colony system, although they have used the simple S-shape and largest gap routing strategies which will be presented in the following section.

5.2 S-shape and largest gap strategy

The S-shape and the largest gap strategies are routing policies which are very frequently used in conjunction with the OBP. In the subsequent brief description of these and other routing strategies, it is assumed that the depot is located in front of the leftmost picking aisle. We refer to Hall (1993) and Roodbergen and de Koster (2001b) for a detailed presentation of the strategies for warehouses with one block and with multiple blocks,

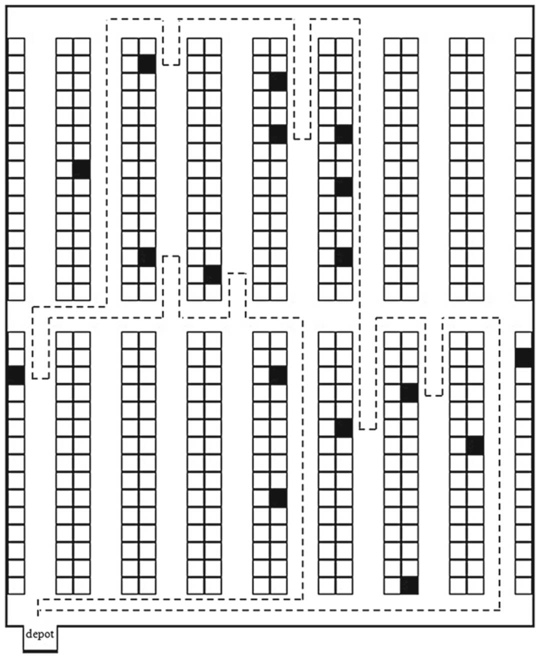


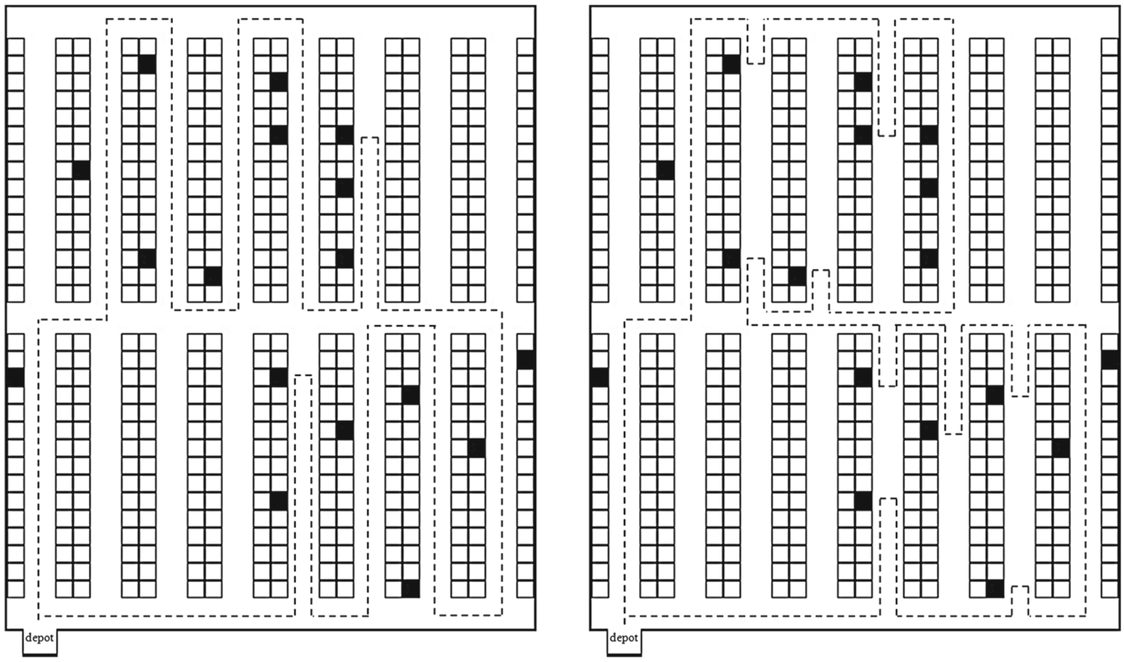
Fig. 4 Example for an optimal picking tour

respectively. The following denotations are used to describe the routing strategies: Let $j_{\min}$ and $j_{\max}$ be the leftmost and the rightmost picking aisle containing at least one requested item, and let $j^i_{\min}$ and $j^i_{\max}$ be the leftmost and rightmost subaisle of block i that has to be visited.

The S-shape (or traversal) routing policy requires that each subaisle containing at least one requested item is traversed entirely. The picker starts at the depot and proceeds to $j^2_{\min}$ by traversing $j_{\min}$ of block #1. Then, from left to right, each subaisle of block #2 containing a requested item is traversed. If the number of subaisles to be traversed in block #2 is odd, the picker returns to the middle cross aisle after picking the items located in $j^2_{\max}$. Having completed all picks in block #2, the picker moves to $j^1_{\max}$ and starts retrieving all items to be collected from block #1 by traversing the corresponding subaisles one by one from the right to the left. Again, it may also be necessary here to apply the return strategy to the last subaisle to be visited. Finally, the order picker returns to the depot.

Note that this algorithm differs slightly from the S-shape routing policy described by Roodbergen and de Koster (2001b). After having changed over from block #2 to block #1, Roodbergen and de Koster (2001b) choose the leftmost subaisle of the first block that still has to be visited instead of $j^1_{\max}$ if this subaisle is closer to $j^2_{\max}$. However, they state that this approach will increase the tour length because it may lead to tours in which the picker visits the subaisles in block #1 from left to right, then has to traverse a considerable part of the front cross aisle before he can return to the depot. Therefore, we use the above-described small modification of their S-shape policy here. An example tour following from our version is depicted in Fig. 5a. The time-complexity function of this routing policy is independent of the number of pick locations and increases linearly with the number of picking aisles.

When the largest gap policy is applied, each subaisle containing at least one requested item is treated in such a way that the non-traversed distance is maximal. This concept leads to the following strategy:



(a) **(b)**

Fig. 5 Picker tour examples, generated by means of **a** the S-shape strategy, **b** the largest gap strategy

The order picker leaves the depot by traversing the first part of $j_{\min}$ and subaisle $j^2_{\min}$. He then moves to $j^2_{\max}$ by entering each subaisle of block #2 from the rear cross aisle up to its largest gap. Subaisle $j^2_{\max}$ is traversed and the middle cross aisle is used to collect the remaining items located in this block. Afterwards, the picker moves to the leftmost subaisle of block #1 that still has to be visited and proceeds from left to right, entering the subaisles of block #1 from the middle cross aisle and proceeding up to their respective largest gaps. When finally $j^1_{\max}$ has been traversed, the picker returns to the depot. On this part of his tour, he collects the remaining items by entering the corresponding subaisles from the front cross aisle.

The time-complexity function of this algorithm is linear in the number of pick locations and picking aisles. Again, this algorithm slightly differs from the description by Roodbergen and de Koster (2001b), where the picker—after having proceeded from block #2 to block #1—is permitted to travel to the rightmost subaisle of block #1 that has to be entered from the middle cross aisle. Our modification leads to tours with fewer changes of movement directions. This is because the largest gap policy by Roodbergen and de Koster (2001b) may generate tours in which the order picker continues to the right after having changed over to block #1, resulting in a corresponding direction change. Now two changes in direction have to be performed in order to apply the largest gap strategy to this block, followed by a further direction change necessary for the return to the depot.

An example for a tour provided by our version of the largest gap strategy is given in Fig. 5b. It becomes evident that this tour can no longer be considered as "simple" and "straightforward". This view particularly holds if the tour is compared to the optimal solution to the same problem depicted in Fig. 4. Thus, we conclude again that for warehouse layouts with two or more blocks no valid argument exists for restricting the routing of the order pickers to solutions provided by simple heuristics like S-shape or largest gap strategy.

5.3 Aisle-by-aisle heuristic

As for the aisle-by-aisle heuristic introduced by Vaughan and Petersen (1999), each picking aisle containing at least one pick location is visited exactly once, i.e. the order picker starts at the depot, picks all requested items in the first picking aisle, then he collects all items in the second aisle and so on. After reaching the rightmost picking aisle that has to be visited, he returns to the depot. Hence, one has only to determine which cross aisle is used to change over from one picking aisle to another. This is determined by dynamic programming.

At first, for each cross aisle $i \in$ {“front”, “middle”, “rear”}, the distance is determined which has to be travelled when the order picker starts at the depot, visits the pick locations in the first picking aisle and exits the first picking aisle by using cross aisle i. Then the second picking aisle is considered and the distance is calculated which the order picker has to travel if he exits picking aisle 1 via cross aisle $i \in$ {“front”, “middle”, “rear”}, picks all requested items located in the second picking aisle and then exits this aisle by using cross aisle $\tilde{i} \in$ {“front”, “middle”, “rear”}. This results in three different distances for each cross aisle $\tilde{i}$. For each cross aisle $\tilde{i}$, the smallest of these three distances is taken as the distance needed to start at the depot, pick all requested items located in the first two picking aisles and exit picking aisle 2 via cross aisle $\tilde{i}$. This procedure is applied to each picking aisle until $j_{\max}$ has been taken into consideration. This results in the distance to be travelled by the order picker when picking all requested items and exiting the rightmost picking aisle through the front cross aisle. Since the picker has to complete his tour at the depot, the distance between $j_{\max}$ and the depot has to be added. An example tour resulting from the aisle-by-aisle heuristic is depicted in Fig. 6a. For this heuristic, each picking aisle has to be considered once and, therefore, the time-complexity function is linear in the number of picking aisles.

5.4 Combined$^+$ heuristic

The combined heuristic introduced by Roodbergen and de Koster (2001b) follows a similar approach as the aisle-by-aisle heuristic does. Here, each subaisle containing at least one requested item is visited exactly once, i.e. each block is considered separately. At first all requested items located in block #2 are collected and then block #1 is considered.

The picker starts at the depot, enters the leftmost picking aisle that contains at least one requested item, and proceeds to the middle cross aisle. Now only block #2 is considered to which the aisle-by-aisle heuristic designed for warehouses with a single-block layout is applied. Note that the picker will not return directly to the depot but complete his tour through block #2 by leaving the last subaisle to be visited in this block via the middle cross aisle. Then the picker proceeds to the rightmost subaisle that has to be visited in block #1. Again, the aisle-by-aisle heuristic for single-block

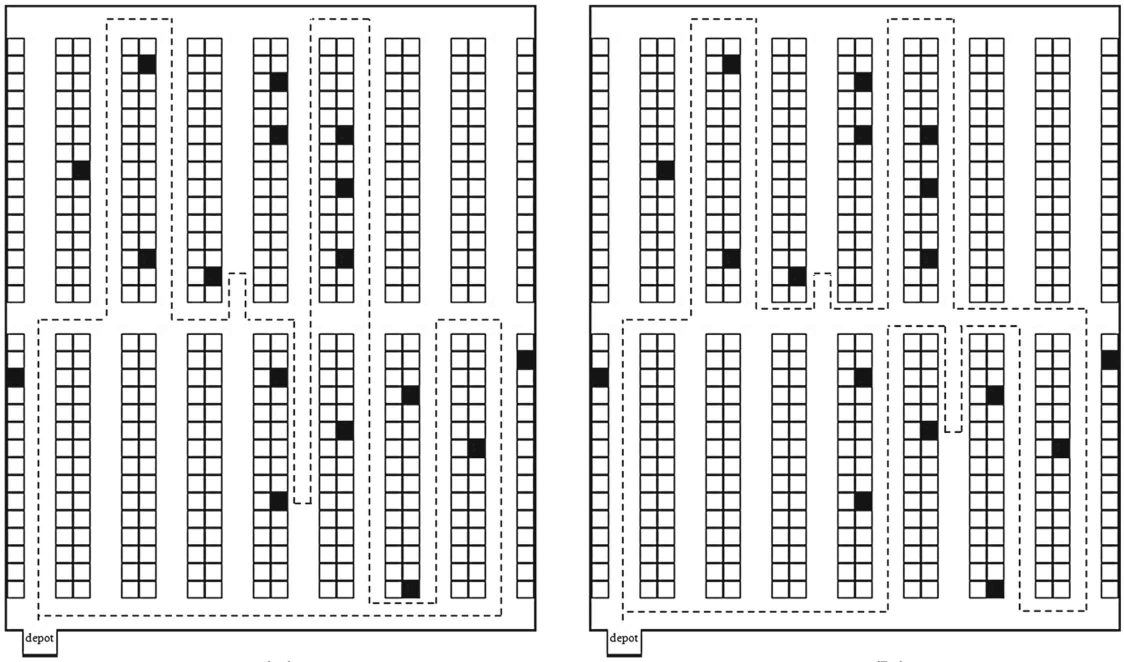


Fig. 6 Picker tour examples, generated by means of **a** the aisle-by-aisle strategy, **b** the combined$^+$ strategy

layouts is applied. In contrast to the original aisle-by-aisle heuristic, the picker has to move from right to left.

In the described procedure, the routing scheme for block #2 includes a move from left to right. Thus, in block #1, the order picker has to traverse the leftmost aisle with pick locations in order to access block #2. However, there may be situations in which it can be advantageous to deviate from this path (e.g. if the storage positions to be visited in the first subaisles of block #1 are located near to the middle cross aisle (Roodbergen and de Koster 2001b)). In this case, the picker may visit several subaisles of block #1 before moving on to block #2. Generally, a tour can be constructed as follows: Let m be the number of picking aisles. The order picker starts at the depot and proceeds to a picking aisle $j^* \in \{1, \ldots, m\}$ by using the front cross aisle. Then the above-described, slightly modified aisle-by-aisle heuristic is applied to the first j^* subaisles of block #1. After picking all requested items in these subaisles, the picker proceeds to block #2 in order to visit all pick locations in this block. (The tour through block #2 is not changed by the modification.) The (modified) aisle-by-aisle heuristic approach is then applied to the last $m - j^*$ subaisles of block #1. Finally, the picker returns to the depot. By optimizing over $j^* \in \{1, \ldots, m\}$ we obtain a tour that is not longer than the tour provided by the original combined heuristic.

The combined heuristic into which this optimization step has been integrated is called combined$^+$ heuristic (Roodbergen and de Koster 2001b). Since the combined heuristic is composed of multiple applications of the aisle-by-aisle heuristic (one application for each block), the time-complexity function for this heuristic also increases linearly with the number of picking aisles. When using the combined$^+$ heuristic, a slightly modified variant of the combined heuristic is applied for each $j^* \in \{1, \ldots, m\}$, generating m different tours. Therefore, we have a quadratic increase of the time-complexity function in the number of picking aisles. An example tour for the combined^{+} heuristic is shown in Fig. 6b.

5.5 A heuristic derived from the exact solution approach

The time-complexity function of the exact approach is linear in the number of picking aisles m. However, it contains a large constant by which m has to be multiplied. Therefore, application of this algorithm to large problem instances encountered in practice is likely to be more time-consuming than the usage of simple routing strategies will be. The constant originates from the number of subgraphs that have to be constructed in the exact solution approach. In each step, i.e. for each picking aisle, $6 + 6 + 14 = 26$ edge configurations have to be added to up to 25 PTSs. This means, if we neglect the fact that the addition of some configurations to certain PTSs cannot lead to an (optimal) order picking tour, up to $26 \cdot 25 = 650$ subgraphs have to be constructed in each iteration of the algorithm.

In order to decrease the large constant in the time-complexity function, we do not always consider PTSs from all equivalent classes but, at some points, only the PTS with the minimum sum of edge weights. To be more precise, all PTSs except the shortest one are deleted after each change of a picking aisle. Based on this idea, a heuristic solution approach can be designed as follows:

Consider the exact algorithm at the iteration before the requested items in picking aisle j are treated, i.e. after having constructed the L_j^--PTSs. Then the shortest of the L_j^--PTSs is determined and configurations (1)–(5) (see Fig. 3) are added to this subgraph. This results in up to five L_j^{+1}-PTSs. Following the procedure of the exact solution approach, configurations are added to each PTS representing the best subgraph of its corresponding equivalence class. Then, the same procedure is applied to the L_j^{+2}-PTSs which leads to L_{j+1}^--PTSs. This transition includes an aisle change and, therefore, all subgraphs, except for the best PTS, are deleted.

Furthermore, as can be derived from the above description, configuration (6) is neglected, i.e. the algorithm will not lead to tours in which a subaisle is traversed twice. This configuration is of course needed to ensure optimality of the tour. (Consider a PRP in which all requested items are located in the first subaisle of block #2. Then, an optimal solution has to contain configuration (6) in the first subaisle of block #1.) However, pretests have shown that only a few cases exist in which this configuration is actually necessary. If this configuration is included, the respective tour often proves itself to be non-optimal. Also, several optimal tours may exist and some can be obtained without using configuration (6).

Of course, optimality of an order picking tour cannot be guaranteed by this approach. Moreover, this heuristic may not lead to a feasible solution at all because it will only result in a few subgraphs and not all L_m^{+2}-PTSs necessarily correspond to a feasible tour. In order to guarantee that at least one feasible tour is obtained, we always consider a subgraph belonging to one of the equivalence classes 3, 4 and 6–9 in Roodbergen and de Koster (2001a). After each step in which the shortest PTS is determined, it is checked whether this subgraph belongs to one of these equivalent classes. If this is not the case, we additionally consider the shortest subgraph belonging to one of these classes. This is done because only eight equivalence classes (classes 2–9) correspond to feasible tours (Roodbergen and de Koster 2001a) and two of these may lead to an infeasible one if they occur as $L^{+1}_{j_{\max}}$-PTSs, where $j_{\max}$ denotes the last picking aisle to be visited.

This modification guarantees feasibility of the solutions provided by the heuristic. However, adding edge configurations to only one instead of 25 subgraphs may lead to a poor solution quality. Especially, in the early stages of the algorithm decisions may be made which result in a poor final solution quality, because, at an early stage, a single edge configuration can have a quite large impact on the sum of edge weights of the subgraph. If a PRP with a small number of requested items per subaisle is considered, then configuration (1) will lead to a larger sum of edge weights than configurations (3)–(5). Hence, the heuristic will tend to select one of these three configurations resulting in subgraphs that are not connected. At a later stage, edge configurations may have to be added to connect the subgraph regardless of the impact on the length of the tour. In order to overcome this problem, our heuristic is combined with the exact solution approach. This means the exact solution approach is applied to the first $s = \lceil p \cdot m \rceil$ picking aisles resulting in several L^{+2}_{s}-PTSs. Then the heuristic solution approach is applied to the last $m - s$ picking aisles. The parameter p denotes the percentage of picking aisles to which the exact solution approach is applied. The larger p is chosen, the higher the computational effort gets, but a larger value of p tends to result in a better solution quality. The choice $p = 1$ results in the exact solution approach. In the appendix, a pseudo-code of this heuristic approach is given in order to point out the differences between the exact algorithm and the heuristic solution approach.

6 An iterated local search for the JOBPRP

For evaluating the impact of the routing strategies on solutions to the JOBPRP, we combine the routing algorithms explained in Sect. 5 with the iterated local search algorithm (ILS) of Henn et al. (2010) for the OBP. The heuristic consists of two alternating phases, namely a local search and a perturbation phase. In the local search phase an initial solution is improved by means of a certain neighborhood structure until a local optimum is reached. In order to overcome local optima, the incumbent solution is partially modified in the perturbation phase and improved in the local search phase again. If the new solution stemming from the local search procedure passes an acceptance criterion, this solution becomes the new incumbent solution. Otherwise, the perturbation phase is applied to the previous solution again.

A general pseudo-code of our algorithm for the JOBPRP is depicted below. For a more detailed presentation of the ILS algorithm we refer to Henn et al. (2010). When dealing with the joint problem, it is sufficient to consider solutions to the OBP and solve the PRP in order to evaluate the solutions. Consequently, we only deal with solutions to the OBP, i.e. each solution x does not correspond to the JOBPRP, but only to the batching (sub)problem. The solution approaches to the PRP are only used for the determination of the corresponding objective function values. Therefore, the following definition regarding the objective function is needed: Let $f_r(x)$ be the objective function value resulting from the solution x to the OBP if the tours are constructed by applying the routing algorithm r. Furthermore, let N^* denote the number of batches in the best known solution.

Algorithm 1 ILS for the JOBPRP

Input: problem data, rearrangement parameter λ, threshold parameter μ, interval t, routing strategy r;

Output: solution x^* to the OBP and corresponding total tour length $f_r\left(x^*\right)$;

generate initial solution x by applying the FCFS rule;
$x^* :=$ local_search (x); $x_{\text{inc}} := x^*$;
repeat
  $x :=$ perturbation $\left(x_{\text{inc}}, \lfloor N^* \cdot \lambda + 1 \rfloor\right)$;
  $x :=$ local_search (x);
  if $f_r(x) < f_r\left(x^*\right)$ **then**
    $x^* := x$; $x_{\text{inc}} := x$;
  end if
  if no improvement of $f_r\left(x^*\right)$ during t **and** $f_r(x) - f_r\left(x^*\right) < \mu \cdot f_r\left(x^*\right)$ **then**
    $x_{\text{inc}} := x$;
  end if
until termination condition is met

An initial (and first best) solution x is determined by applying the first-come-first-served (FCFS) rule. Then the local search procedure is used in combination with the routing strategy r in order to improve x. Afterwards, the perturbation and the local search phase alternate until a termination condition is met. In the local search procedure two different types of neighborhoods are used. With respect to the first neighborhood structure, two solutions are called neighbors if one solution can be obtained from the other by interchanging two orders of the batches (SWAP). The second structure considers solutions that can be generated by assigning a single order to a different batch (SHIFT). The local search phase stops when no further improvement is possible by means of these neighborhood structures. The perturbation phase constructs a new solution by interchanging a random number of orders of two batches. An iteration of this phase can be described as follows: Two batches α and β and an integer number υ between 1 and half of the number of orders in α and β are randomly selected. The first υ orders are removed from both α and β. Then the υ orders from α are assigned to β and vice versa as long as the capacity constraints are not violated. The remaining orders are assigned to a new batch. This iteration is repeated $\lfloor N^* \cdot \lambda + 1 \rfloor$ times, i.e. the more batches are contained in the best known solution, the more iterations are performed in the perturbation phase.

7 Numerical experiments

7.1 Setup

In order to evaluate the performance of the algorithms for the PRP and the JOBPRP, thorough numerical experiments are carried out. The generation of the test instances is partially based on the problem sets in Henn et al. (2010) and Henn and Wäscher

(2012). We do not use the same problem instances here because they considered the OBP in a warehouse with a single-block layout, whereas in this paper a two-block layout is assumed. Furthermore, some modifications are made due to the fact that the solution quality of routing strategies is dependent on the dimension of the warehouse.

We consider a warehouse with a two-block layout and 50 storage locations in each subaisle (25 locations on each side of the subaisle). Within each subaisle, we assume two-sided picking, i.e. when a picker is located in the center of a subaisle, he is able to collect items from both sides without additional movements. A single storage location has a length of 1 length unit (LU) and a width of 1.5 LUs. The distance between two opposite storage locations belonging to the same subaisle amounts to 2 LUs. In addition, the picker has to move 1 LU from the first or the last storage location of a subaisle to reach a cross aisle. Therefore, a picker has to travel 46 LUs when traversing a subaisle and 5 LUs when changing from one aisle to another. Since we aim at evaluating the performance of routing strategies and since the performance of the algorithms is known to be dependent on the number of picking aisles, in contrast to Henn and Wäscher (2012) (who fixed the number of aisles to 10), we consider problem instances with either 10, 20 or 30 picking aisles for the PRP and either 10 or 30 aisles for the JOBPRP which results in warehouses with 1000, 2000 and 3000 storage locations, respectively.

In our numerical experiments we consider problem instances of four different sizes: 20, 40, 60 and 80 customer orders. The number of requested items in a customer order is uniformly distributed over the set $\{5, \ldots, 25\}$. The capacity of the picking device, which is the maximum number of items that can be collected in a single tour, amounts to 30, 45, 60 and 75. This means that the expected maximum number of customer orders included in one tour varies between 2 and 5. Of course, when dealing with the PRP, the number of orders is set to 1 and the capacity of the picking device is not smaller than the number of requested items. In the experiments for the PRP, the number of pick locations has been fixed to 30, 45, 60 and 75. Since the solution quality of the solution approaches to the PRP is only investigated in order to decide which of these algorithms should be combined with the iterated local search approach, we limit the considerations to a random storage assignment, i.e. the probability of each article to be contained in a customer order is identical for all articles. This kind of storage assignment has also been used by Albareda-Sambola et al. (2009) and Henn et al. (2010). For the JOBPRP, we also consider the more sophisticated class-based storage assignment policy used by Henn et al. (2012). The general idea of this policy is to assign articles with high demand frequencies to storage locations near to the depot. In the approach of Henn et al. (2012), the articles are, therefore, divided into three classes A, B and C. 10% of all articles with the highest demand frequency represent up to 52% of the total demand (class A) and 30% are responsible for 36% of the demand (class B). The remaining articles have quite low demand frequencies and are assigned to class C. Within each class, articles are randomly assigned to a storage location in a certain subaisle. The determination of this subaisle is based on the distance between the subaisle and the depot; class A (C) articles are assigned to a storage location in a subaisle representing 10% (60%) of all subaisles with the shortest (longest) distance to the depot.

Table 1 Average deviation of the objective function values obtained by the heuristic routing strategies from the optimal objective function values in %

m	n	SS	LG	AA	C^{+}	HMEA
10	30	20.2	19.3	15.6	6.8	**4.0**
10	45	15.2	22.2	13.3	5.1	**3.2**
10	60	9.6	23.4	8.8	3.7	**2.3**
10	75	8.9	26.5	9.2	3.2	**2.9**
20	30	27.2	22.3	16.3	7.2	**5.8**
20	45	24.4	22.1	16.4	6.9	**4.5**
20	60	20.8	23.7	14.5	6.4	**3.1**
20	75	18.4	23.5	14.2	6.6	**2.7**
30	30	27.4	27.7	16.3	**7.0**	7.6
30	45	26.4	26.3	16.5	7.3	**5.4**
30	60	25.0	24.3	16.7	7.0	**4.3**
30	75	23.9	25.1	16.1	7.0	**3.8**
Average		20.6	23.9	14.5	6.2	**4.1**

A combination of the above-mentioned parameter values results in 12 problem classes for the PRP and 64 classes for the JOBPRP. For each PRP class, 100 test instances have been generated and for each class of the JOBPRP, 50 instances have been considered, leading to 4000 problem instances. The computations have been carried out on a desktop PC with a 3.4 GHz Pentium processor and 8 GB RAM. All algorithms have been encoded in C++ using Microsoft Visual Studio 2013. Before the JOBPRP is considered, the solution quality of the algorithms for the PRP is investigated.

7.2 Picker Routing Problem

In this section, the performance of the solution approaches to the PRP presented in Sect. 5 is evaluated. The new heuristic approach introduced in Sect. 5.5 includes a parameter p which has to be specified. According to results from pretests we have chosen $p = 0.25$. The results from the numerical experiments for the PRP are depicted in Table 1. Each instance has been solved by each of the algorithms in less than one second. Computing times, therefore, are not reported in greater detail here.

For each problem class, Table 1 depicts the average deviation of the objective function values obtained by the routing strategies under discussion (SS: S-shape; LG: largest gap; AA: aisle-by-aisle; C^{+}: combined^{+}; HMEA: heuristically modified exact algorithm, where the best equivalence class of an iteration is considered only) from the optimal values that were obtained by the algorithm of Roodbergen and de Koster (2001a). The problem classes are identified by the number of picking aisles m and the number of pick locations n. For each problem class, the number in bold represents the best result obtained.

The results presented in Table 1 demonstrate that the HMEA outperforms the other routing strategies and generates the shortest tours. Furthermore, the solution quality improves with an increasing number of pick locations for the following reason: The exact algorithm as well as the HMEA do not explicitly consider each pick location but rather each subaisle. Let us consider the configurations added to collect requested items in a subaisle. The sum of the edge weights that results from adding configuration (1) or (6) (see Fig. 3) is independent from the requested items. In contrast to that, the sum of the edge weights for configurations (3)–(5) is strongly dependent on the location of the requested items within the subaisle. Therefore, the sum of edge weights may vary very strongly for a small number of requested items, but for a large number of pick locations in a subaisle, it converges to the sum of the edge weights that results from adding configuration (6). Hence, several equivalence classes may lead to a near optimal solution and the impact of deleting an equivalence class, which would lead to an optimal tour, is not as large as it is for problems with a small number of requested items. In addition, it can be seen that the number of picking aisles also has an impact on the solution quality of the HMEA. This is also due to the effect explained above because an increasing number of picking aisles results in a decreasing number of pick locations per subaisle.

The S-shape heuristic, which represents the most frequently used routing policy in practice (Roodbergen 2001), leads to poor results. For problem instances with a small number of picking aisles and a large number of pick locations, the S-shape heuristic provides acceptable tours because if there are a lot of requested items in each subaisle, the additional length to traverse each subaisle through the entire length compared to the distance that is needed to collect the requested items in a subaisle will be small. However, the solution quality deteriorates with an increasing number of picking aisles, where the S-shape heuristic leads to average deviations of up to 27%. This observation coincides with the results from numerical experiments done by Roodbergen and de Koster (2001b). Over all problem classes, the average deviation amounts to 20.6%, which is more than 16 percentage points higher than the deviation of the objective function value obtained by the HMEA from the optimal objective function value.

The largest gap heuristic provides an average deviation from the optimal total tour length of 23.9%. When applied to a PRP in a warehouse with a single-block layout, it is known that the solution quality of this heuristic improves with a decreasing number of pick locations because, in this case, the distances between adjacent pick locations within the same subaisle (called gaps) increase. Since tours constructed by this routing strategy do not contain the part of an aisle that corresponds to the largest gap, larger gaps lead to a better solution. This argumentation matches with the results from the instances containing 10 picking aisles. However, if we take a look at the results corresponding to problem classes containing 30 aisles, we will see the opposite: The solution quality improves with an increasing number of pick locations. The problem instances containing 30 picking aisles are characterized by a quite small number of pick locations per subaisle. This number ranges from 0.5 to 1.25. Therefore, it can be expected that the gaps in the subaisles are quite large. Thus, the largest gap policy leads to very short distances to be traveled in subaisles. The distances to be traveled in cross aisles, however, may be quite large. In a worst case scenario, the middle cross aisle has to be traversed twice which results in large tour lengths for problems with a large number of picking aisles. If the number of pick locations increases, the length of an order picking tour will also increase, especially due to the long distances to be traveled in the subaisles. Therefore, the percentage of the distance to be traveled in cross aisles decreases and the large distance resulting from the movements in the middle cross aisle has a minor impact on the objective function value, resulting in a smaller deviation from the optimal objective function value. However, if the number of pick locations increases, the deviation from the optimal objective function value will also increase at some point due to the increasing deviation resulting from the movements within the subaisles.

Across all problem classes, the aisle-by-aisle heuristic leads to an average deviation of 14.5% and also results in quite long tours and is outperformed by the combined^{+} heuristic. This is a consequence of the fact that the combined^{+} heuristic considers the problem blockwise, i.e. only subaisles are considered instead of complete picking aisles. Therefore, the combined^{+} heuristic leads to an acceptable solution quality even for problem instances in which the number of pick locations per subaisle is quite small. Nevertheless, the solution quality deteriorates with a decreasing number of pick locations per subaisle because this heuristic is a combination of the S-shape and the return strategy and, as mentioned before, the S-shape policy leads to very poor results in these cases.

In summary, it can be stated that the HMEA leads to the smallest deviation from the optimal objective function value. Furthermore, we can see that the combined^{+} heuristic also results in rather short order picking tours. However, if only single instances of the PRP have to be solved, the exact approach should be applied because the computing times are below one second and, therefore, negligible. Computing times become an issue, though, when instances of the PRP have to be solved repeatedly within the proposed ILS approach to the JOBPRP, as will be demonstrated in the next subsection.

7.3 Joint Order Batching and Picker Routing Problem

7.3.1 Selection of routing strategies and parameters for the ILS

In order to deal with the OBP and the PRP in a more integrated way, the exact solution approach of Roodbergen and de Koster (2001b) and the routing strategies which have been shown to perform quite well in the previous subsection have been combined with the iterated local search approach presented in Sect. 6, namely the S-shape and largest gap policy, the combined^{+} heuristic and the HMEA. As termination criterion for our solution approach, we use a time limit T which is measured in seconds and defined by $T = 3N$. By definition, the time limit is only dependent on the number of orders N, i.e. on the problem size of the OBP and not on the routing strategy used. Thus, a tradeoff exists between considering a larger part of the solution space of the OBP by running a large number of iterations in the iterated local search and constructing good tours. This can also be seen from Table 2.

In this table, the number of iterations performed by the ILS is presented in dependency of the number of picking aisles m, the number of orders N and the routing strategy. An iteration includes two consecutive local search and perturbation phases. Of course, the number of iterations decreases with an increasing number of picking aisles and orders. However, the number of iterations also decreases significantly if more complex routing algorithms are used. This is due to the fact that a large number

Table 2 Average number of iterations performed by the ILS algorithm combined with different routing algorithms ($C = 30$)

m	N	SS	LG	C+	HMEA	Exact
10	20	9220	3822	762	292	181
10	40	3145	1058	201	96	60
10	60	1251	455	116	34	21
10	80	717	206	36	12	7
30	20	3481	1861	296	122	81
30	40	1117	543	66	29	20
30	60	376	174	20	11	6
30	80	209	97	9	6	3

Table 3 Average deviation of the objective function values obtained by the ILS in combination with the exact routing algorithm from the optimal values for $N = 20$ (%)

m	C	Storage assignment	
		Random	Class based
10	30	0.23	0.51
10	45	0.25	0.49
30	30	0.84	0.14
30	45	0.48	0.23

of PRPs has to be solved in each local search phase in order to evaluate the solutions. Although only fractions of a second are needed to solve a single PRP to optimality, the routing algorithm used in the ILS has a large impact on the number of different solutions to the OBP that can be considered. Therefore, it will have to be investigated if this small number of iterations still results in reasonably good solutions to the JOBPRP.

The iterated local search approach contains different parameters which have to be chosen. Henn et al. (2010) used this ILS algorithm to deal with the OBP while the S-shape or the largest gap strategies have been used to evaluate the solutions. They have shown that this approach leads to high quality solutions if the parameters are set as follows: $\lambda = 0.3$, $\mu = 0.05$, $t = \frac{T}{10}$. However, as can be seen from Table 2, the number of iterations within the ILS is strongly dependent on the routing strategy used, and it cannot be guaranteed that these parameter values are also appropriate when dealing with the JOBPRP. We, therefore, used the model formulation presented in Sect. 4 in order to solve small instances ($N = 20$) to optimality and compared the minimum total tour lengths to the objective function values obtained from the ILS approach combined with the exact routing algorithm. The results are depicted in Table 3.

The size of the model formulation rapidly increases with an increasing number of orders N and a growing capacity C. Due to memory restrictions, we were only able to solve all instances of the problem classes with 20 orders and a capacity of up to 45 items. As stated in Sect. 7.1, we considered problem classes with 10 and 30 picking aisles as well as random and class-based storage assignment. Each problem class includes 50 instances which have been solved by integrating the exact routing algorithm into the ILS approach. On average, the objective function value obtained deviates less than 1% from the minimum total tour length. This observation holds for all problem classes under consideration. Thus, we conclude that this approach leads to very good solutions and the parameter selection of Henn et al. (2010) is also appropriate for the JOBPRP.

In the following two subsections, the solution quality of the ILS approach in combination with the different routing algorithms is investigated. We first consider problem instances based on a random assignment of articles to storage locations. Then, instances related to a class-based storage assignment are considered in order to study the impact of the location assignment strategy on the performance of the different solution approaches.

7.3.2 Results for random storage assignment

For each problem class, the average deviation of the objective function values obtained from the application of the ILS in combination with a routing heuristic from the objective function value of the ILS with the exact routing algorithm (denoted by ILS-Exact) is depicted in Table 4. Due to the fact that all deviations are positive, we can conclude that the ILS in combination with the exact routing algorithm outperforms the combintions of the ILS with the other heuristic routing strategies.

When dealing with the OBP, usually the S-shape or the largest gap strategy are used in order to evaluate solutions (Albareda-Sambola et al. 2009; Henn et al. 2010; Henn and Wäscher 2012). The usage of these simple routing strategies, however, results in objective function values far above the objective function values that can be obtained by combining a metaheuristic for the OBP with the exact routing algorithm. On average, across all problem classes, the common integration of the S-shape and the largest gap strategy into the batching algorithm leads to tours that are 17.95 and 22.48% longer than tours obtained by ILS-Exact. If the number of picking aisles is small ($m = 10$) and the capacity of the picking device is large ($C = 75$), then the integration of the S-shape policy leads to deviations that are below 10%. This is because a larger capacity results in batches containing more items and due to the fact that the S-shape strategy leads to good solutions if the number of pick locations per aisle is quite large. However, what concerns all other problem classes, both the S-shape and the largest gap strategy result in very poor solutions. This is particularly true if the number of picking aisles is large ($m = 30$). The integration of the more complex routing algorithms combined^{+} and, in particular, the HMEA leads to acceptable results with an average deviation of 4.87 and 2.64%, respectively.

The results from Table 4 demonstrate that it is pivotal to take the PRP into account when dealing with the OBP. Although the integration of more complex routing algorithms results in a significant lower number of iterations conducted by the batching algorithm, thoroughly dealing with the arising PRPs leads to significant savings with respect to the total tour length. However, the routing approach which should be used is dependent on the computing time which is available for solving the JOBPRP. In our numerical experiments, we considered quite low computing times varying between 1 and 4 min. [Note that some solution approaches to the OBP require up to 20 min for solving instances with 60 customer orders (Henn et al. 2010).] If the time limit of our approach is further increased the number of iterations in the ILS will also increase. Since the number of iterations is a critical issue for complex routing algorithms inte-

Table 4 Average deviation from the objective function value obtained by the ILS in combination with the exact routing algorithm for random storage assignment (%)

m	N	C	ILS			
			SS	LG	C^+	HMEA
10	20	30	20.28	17.90	6.37	**3.72**
10	20	45	14.93	20.21	4.85	**1.64**
10	20	60	12.21	22.72	4.21	**1.18**
10	20	75	9.93	24.68	3.63	**0.88**
10	40	30	19.29	17.56	5.92	**3.44**
10	40	45	13.77	19.32	4.26	**1.68**
10	40	60	10.61	21.88	3.79	**1.35**
10	40	75	8.22	24.09	2.89	**1.20**
10	60	30	18.44	17.68	5.23	**2.76**
10	60	45	13.45	19.51	3.83	**1.72**
10	60	60	10.10	21.30	2.87	**1.21**
10	60	75	7.88	23.61	2.45	**0.85**
10	80	30	18.31	17.49	5.39	**2.86**
10	80	45	13.30	19.38	4.11	**1.74**
10	80	60	10.23	21.40	3.51	**1.57**
10	80	75	7.85	23.56	2.82	**1.52**
30	20	30	24.10	25.52	5.94	**4.95**
30	20	45	24.55	24.49	5.94	**3.18**
30	20	60	23.45	24.56	5.87	**2.57**
30	20	75	22.22	24.39	5.58	**2.11**
30	40	30	24.21	25.03	5.87	**4.73**
30	40	45	24.18	23.98	5.73	**3.44**
30	40	60	22.87	24.08	5.80	**3.15**
30	40	75	21.22	24.02	5.46	**2.48**
30	60	30	23.46	24.89	5.67	**4.76**
30	60	45	23.15	24.01	5.56	**3.86**
30	60	60	22.49	23.57	5.00	**3.50**
30	60	75	21.52	23.01	5.13	**2.42**
30	80	30	23.37	24.62	5.77	**4.30**
30	80	45	22.60	23.35	5.30	**3.22**
30	80	60	21.87	23.73	5.64	**3.31**
30	80	75	20.39	23.68	5.38	**3.14**
Average			17.95	22.48	4.87	**2.64**

grated into the ILS, it can be expected that the solution quality will further improve in this case. Thus, it would get even more important to use a sophisticated routing algorithm in order to solve the arising PRPs. On the other hand, if less computing time is available, it may not be possible to apply the exact routing algorithm within the ILS. In this case, the newly proposed HMEA should be applied to deal with the PRPs since this heuristic allows for more ILS iterations and still provides very good solutions to the PRP.

Table 5 Average deviation from the objective function value obtained by the ILS in combination with the exact routing algorithm for class based storage assignment [%]

m	N	C	ILS			
			SS	LG	C^+	HMEA
10	20	30	16.06	13.41	5.02	**3.76**
10	20	45	11.17	16.40	3.42	**2.11**
10	20	60	9.65	19.43	2.79	**1.47**
10	20	75	8.86	21.61	2.83	**1.27**
10	40	30	15.40	13.92	5.06	**3.78**
10	40	45	10.43	14.74	3.35	**1.98**
10	40	60	8.29	18.88	2.57	**1.26**
10	40	75	7.10	20.94	2.13	**0.98**
10	60	30	14.50	13.31	4.70	**3.31**
10	60	45	9.21	14.74	2.68	**1.50**
10	60	60	7.80	17.57	2.84	**1.30**
10	60	75	8.47	24.52	3.04	**1.38**
10	80	30	14.87	12.99	4.82	**3.83**
10	80	45	9.99	14.97	3.18	**2.14**
10	80	60	8.01	17.15	2.54	**1.62**
10	80	75	6.84	19.36	1.97	**1.48**
30	20	30	23.76	27.41	6.02	**5.51**
30	20	45	22.59	26.46	5.53	**3.27**
30	20	60	21.17	26.29	5.24	**2.92**
30	20	75	19.36	26.53	4.70	**2.35**
30	40	30	21.72	25.82	5.48	**4.36**
30	40	45	20.77	25.08	5.10	**3.58**
30	40	60	19.39	25.25	4.91	**2.99**
30	40	75	18.08	25.60	4.60	**2.26**
30	60	30	22.17	25.73	5.54	**4.56**
30	60	45	20.97	25.03	5.05	**3.35**
30	60	60	19.71	25.25	5.02	**3.08**
30	60	75	18.12	25.60	4.84	**2.96**
30	80	30	21.80	25.94	5.47	**4.49**
30	80	45	20.66	25.06	4.99	**3.36**
30	80	60	18.97	24.96	4.46	**3.00**
30	80	75	17.32	25.08	4.19	**2.67**
Average			15.41	21.46	4.19	**2.75**

7.3.3 Results for class-based storage assignment

In this section, the results for class-based storage assignment are reported. Again, the different, already previously discussed solution approaches are compared to ILS-Exact. Table 5 depicts the corresponding deviations of the objective function values that were obtained for the different problem classes.

In general it can be noted that the results are very similar to those for random storage assignment. It can be noted again that the integration of the simple S-shape and largest gap strategies lead to very poor results. Compared to the results for random storage assignment, the deviation from the total tour lengths obtained by ILS-Exact only slightly decreases. The average deviation is decreased by 2.54 (S-shape strategy) and 1.02 (largest gap strategy) percentage points. As before, the S-shape policy leads to very long tours if the ratio m/C is quite large. However, for problem instances with a small number of picking aisles ($m = 10$) and a small capacity of the picking device ($C = 30$), the deviation from the total tour lengths obtained by ILS-Exact is significantly reduced compared to the case of random storage assignment. Since articles with high demand frequencies are assigned to the subaisles with a short distance from the depot, only few items have to be retrieved that are located in more distant subaisles. This leads to tours with fewer subaisles to be visited, which is advantageous when applying the S-shape policy. When considering the results for the largest gap strategy, we can observe different effects of the class-based storage assignment. For instances with a small number of picking aisles ($m = 10$), the largest gap heuristic leads to smaller deviations compared to the case of random storage assignment. This can also be explained by the fact that a smaller number of subaisles has to be visited. However, the deviations even increase for a large number of picking aisles ($m = 30$). This is due to the following two reasons: First, the main drawback of the largest gap strategy in a two-block layout consists in traversing the middle cross aisle twice which results in very long tours when considering a large number of picking aisles. Second, this heuristic only leads to good solutions if the gaps are quite large. When applying a class-based storage assignment procedure to a large warehouse, a considerably large number of subaisles is assigned to articles with a quite high demand frequency. Therefore, lots of items may be picked in these subaisles resulting in very small gaps. This effect is not compensated by the fact that the other subaisles only contain very few requested items.

Regarding the integration of the more sophisticated routing heuristic combined$^+$ and the HMEA, we can again observe that these approaches result in a good solution quality with an average deviation of 4.19% (combined$^+$ heuristic) and 2.75% (HMEA) which is very similar to the results in the case of random storage assignment.

Considering the average over all problem classes, for each solution approach, the deviation in the case of class-based storage assignment differs less than 0.5 percentage points from the deviation in the case of randomly assigned articles. We, therefore, conclude that our results hold for both random and more sophisticated storage assignment procedures. Furthermore, all deviations remain positive which means that again, the integration of the exact routing algorithm outperforms the application of heuristic routing strategies.

8 Conclusion and outlook

In this paper, we have dealt with the Order Batching and the Picker Routing Problem which are pivotal for operating manual order picking systems efficiently. Although both problems are closely interconnected, most approaches only deal with the Order Batching Problem and apply simple routing policies for solving the arising Picker

Routing Problems. The intention of this paper was to investigate whether the solution quality can be improved by solving the two problems in a more integrated way. Therefore, an exact routing algorithm as well as several heuristic routing strategies have been combined with an iterated local search algorithm for the Order Batching Problem. Furthermore, a new solution approach derived from an exact routing algorithm has been designed in order to close the gap between the complex exact routing algorithm and the simple routing heuristics. Based on extensive numerical experiments with fixed computing times, the performance of the routing algorithms as well as the performance of the iterated local search approach combined with different routing algorithms have been evaluated.

The results from the numerical experiments have shown that common approaches are outperformed by far. On average, the resulting tours turned out to be up to 25% longer than the tours obtained by the iterated local search into which an exact routing algorithm has been integrated. It has been demonstrated that— at least for medium computing times available— much more attention should be given to the arising Picker Routing Problems. If computing time is not a critical issue it is pivotal to find (near) optimal solutions to the routing problems. Even if solutions have to be provided within a very small amount of computing time, the routing problems should be taken into account. In this case, either the newly proposed heuristically modified exact approach or the combined^{+} heuristic are to be applied.

The observations made in this paper also hold for warehouses with layouts composed of more than two blocks. Since the solution quality of common routing strategies tends to deteriorate with an increasing number of blocks and, furthermore, the tours generated get more complex, the consideration of the arising Picker Routing Problems gets even more important. However, no efficient exact algorithm is available for Picker Routing Problems in warehouses with more than two blocks which is why other approaches have to be considered. The integration of sophisticated TSP algorithms such as the Lin-Kernighan-Helsgaun Heuristic could be a very promising approach.

Further research could also concentrate on the consideration of picker blocking aspects because congestion is an important topic in warehouses with narrow aisles and the savings reached by the integration of the exact routing algorithm may diminish if this leads to picker blocking on a larger scale. From a practical point of view it would also be reasonable to consider the on-line variant of the Order Batching and Picker Routing Problem in which not all customer orders are known in advance.

References

Albareda-Sambola M, Alonso-Ayuso A, de Blas C (2009) Variable neighborhood search for order batching in a warehouse. Asia Pac J Oper Res 26:655–683

Bozer YA, Kile JW (2008) Order batching in walk-and-pick order picking systems. Int J Prod Res 46:1887–1909

Clarke G, Wright JW (1964) Scheduling of vehicles from a central depot to a number of delivery points. Oper Res 12:568–581

Coyle JJ, Bardi EJ, Langley CJ (1996) The management of business logistics, 6th edn. West Publishing Company, St. Paul de Koster R, Roodbergen KJ, van Voorden R (1999a) Reduction of walking time in the Center of de Bijenkorf. In: Speranza MG, Stähly P (eds) New trends in distribution logistics. Springer, Berlin, pp 215–234

de Koster R, van der Poort E, Wolters M (1999b) Efficient orderbatching methods in warehouses. Int J Prod Res 37:1479–1504

de Koster R, Le-Duc T, Roodbergen KJ (2007) Design and control of warehouse order picking: a literature review. Sci Dir 182:481–501

Elsayed EA (1981) Algorithms for optimal material handling in automatic warehousing systems. Int J Prod Res 19:525–535

Elsayed EA, Unal OI (1989) Order batching algorithms and travel-time estimation for automated storage/retrieval systems. Int J Prod Res 27:1097–1114

Frazelle E (2002) World-class warehouse and material handling. McGrawHill, New York

Gademann N, van de Velde S (2005) Order batching to minimize total travel time. IIE Trans 37:63–75

Gibson DR, Sharp GP (1992) Order batching procedures. Eur J Oper Res 58:57–67

Goetschalckx M, Ratliff HD (1988) Order picking in an aisle. IIE Trans 20:53–62

Grosse EH, Glock CH, Ballester-Ripoll R (2014) A simulated annealing approach for the joint order batching and order picker routing problem with weight restrictions. Int J Oper Quant Manag 20:65–83

Hall RW (1993) Distance approximations for routing manual pickers in a warehouse. IIE Trans 25:76–87

Helsgaun K (2000) An effective implementation of the Lin–Kernighan traveling salesman heuristic. Eur J Oper Res 126:106–130

Henn S, Wäscher G (2012) Tabu search heuristics for the order batching problem in manual order picking systems. Eur J Oper Res 222:484–494

Henn S, Koch S, Dörner K, Strauss C, Wäscher G (2010) Metaheuristics for the order batching problem in manual order picking systems. Bus Res (BuR) 3:82–105

Henn S, Koch S, Wäscher G (2012) Order batching in order picking warehouses: a survey of solution approaches. In: Manzini R (ed) Warehousing in the global supply chain: advanced models, tools and applications for storage systems. Springer, London, pp 105–137

Kulak O, Sahin Y, Taner ME (2012) Joint order batching and picker routing in single and multiple-cross-aisle warehouses using cluster-based tabu search algorithms. Flex Ser Manuf J 24:52–80

Petersen CG (1997) An evaluation of order picking routeing policies. Int J Oper Prod Manag 17:1098–1111

Petersen CG, Schmenner RW (1999) An evaluation of routing and volume-based storage policies in an order picking operation. Decis Sci 30:481–501

Ratliff HD, Rosenthal AR (1983) Order-picking in a rectangular warehouse: a solvable case of the traveling salesman problem. Oper Res 31:507–521

Roodbergen KJ (2001) Layout and routing methods for warehouses. Rotterdam, Trial

Roodbergen KJ, de Koster R (2001a) Routing order pickers in a warehouse with a middle aisle. Eur J Oper Res 133:32–43

Roodbergen KJ, de Koster R (2001b) Routing methods for warehouses with multiple cross aisles. Int J Prod Res 39:1865–1883

Theys C, Bräysy O, Dullaert W, Raa B (2010) Using a TSP heuristic for routing order pickers in warehouses. Eur J Oper Res 200:755–763

Tsai C-Y, Liou JJH, Huang T-M (2008) Using a multiple-GA method to solve the batch picking problem: considering travel distance and order due time. Int J Prod Res 46:6533–6555

Vaughan TS, Petersen CG (1999) The effect of warehouse cross aisles on order picking efficiency. Int J Prod Res 37:881–897

Wäscher G (2004) Order picking: a survey of planning problems and methods. In: Dyckoff H, Lackes R, Reese J (eds) Supply chain management and reverse logistics. Springer, Berlin, pp 323–347

Appendix

Pseudo-codes for the exact algorithm and the heuristic derived from this approach

Algorithm 2 Pseudo-code for the algorithm by Roodbergen & de Koster (2001a)

Input: problem data (containing number of picking aisles m);

compute sum of edge weights for each configuration and picking aisle;
construct L_1^{+1}-PTS by adding configurations for subaisle 1 of block #1 to an empty graph;
for equivalence classes $i = 1$ to 25 **do**
    determine the L_1^{+1}-PTS of class i with the smallest sum of edge weights;
end for
for equivalence classes $i = 1$ to 25 **do**
    construct L_1^{+2}-PTS by adding configurations for subaisle 1 of block #2 to L_1^{+1}-PTS of class i;
end for
for equivalence classes $i = 1$ to 25 **do**
    determine the L_1^{+2}-PTS of class i with the smallest sum of edge weights;
end for
for picking aisles $j = 2$ to m **do**
    for equivalence classes $i = 1$ to 25 **do**
        construct L_j^{-}-PTS by adding configurations for movements between picking aisles $j-1$ and j to L_{j-1}^{+2}-PTS of class i;
    end for
    for equivalence classes $i = 1$ to 25 **do**
        determine the L_j^{-}-PTS of class i with the smallest sum of edge weights;
    end for
    for equivalence classes $i = 1$ to 25 **do**
        construct L_j^{+1}-PTS by adding configurations for subaisle j of block #1 to L_j^{-}-PTS of class i;
    end for
    for equivalence classes $i = 1$ to 25 **do**
        determine the L_j^{+1}-PTS of class i with the smallest sum of edge weights;
    end for
    for equivalence classes $i = 1$ to 25 **do**
        construct L_j^{+2}-PTS by adding configurations for subaisle j of block #2 to L_j^{+1}-PTS of class i;
    end for
    for equivalence classes $i = 1$ to 25 **do**
        determine the L_j^{+2}-PTS of class i with the smallest sum of edge weights;
    end for
end for
out of classes $2, 3, \ldots, 9$, determine the L_m^{+2}-PTS with the smallest sum of edge weights;

Algorithm 3 Pseudo-code for the heuristic derived from the exact solution approach

Input: problem data (containing number of picking aisles $\tilde{m}$), percentage p;

compute sum of edge weights for each configuration and picking aisle;
apply the algorithm by Roodbergen & de Koster (2001a) with $m := \lceil p \cdot \tilde{m} \rceil$;
for pickings aisles $j = \lceil p \cdot \tilde{m} \rceil + 1$ to $\tilde{m}$ **do**
    for equivalencees class $i = 1$ to 25 **do**
        construct L_j^--PTS by adding configurations for movements between picking aisles $j-1$ and j to L_{j-1}^{+2}-PTS of class i;
    end for
    determine the L_j^--PTS (denoted by L_j^*) with the smallest sum of edge weights;
    construct L_j^{+1}-PTS by adding configurations for subaisle j of block #1 to L_j^*;
    if L_j^* does not belong to any of the classes $3, 4, 6, 7, 8$ or 9 **then**
        out of classes $3, 4, 6, 7, 8$ and 9, determine the L_j^--PTS (denoted by L_j^{**}) with the smallest sum of edge weights;
        construct additional L_j^{+1}-PTS by adding configurations for subaisle j of block #1 to L_j^{**};
    end if
    for equivalence classes $i = 1$ to 25 **do**
        determine the L_j^{+1}-PTS of class i with the smallest sum of edge weights;
    end for
    for equivalence classes $i = 1$ to 25 **do**
        construct L_j^{+2}-PTS by adding configurations for subaisle j of block #2 to L_j^{+1}-PTS of class i;
    end for
    for equivalence classes $i = 1$ to 25 **do**
        determine the L_j^{+2}-PTS of class i with the smallest sum of edge weights;
    end for
end for
out of classes $2, 3, \ldots, 9$, determine the L_m^{+2}-PTS with the smallest sum of edge weights;

Part VII:

Order Batching and Picker Routing for the Minimization of the Total Tardiness

European Journal of Operational Research 263 (2017) 461–478

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Production, Manufacturing and Logistics

Order picking with multiple pickers and due dates – Simultaneous solution of Order Batching, Batch Assignment and Sequencing, and Picker Routing Problems

André Scholz[a,*], Daniel Schubert[a], Gerhard Wäscher[a,b]

[a] Faculty of Economics and Management, Otto-von-Guericke University Magdeburg, Magdeburg 39106, Germany
[b] School of Mechanical, Electronic and Control Engineering, Beijing Jiaotong University, Beijing 100044, China

ARTICLE INFO

Article history:
Received 22 July 2016
Accepted 20 April 2017
Available online 27 April 2017

Keywords:
Traveling salesman
Order batching
Batch sequencing
Picker routing
Variable neighborhood descent

ABSTRACT

In manual picker-to-part order picking systems, human operators (order pickers) walk or ride through the warehouse, retrieving items from their storage locations in order to satisfy a given demand specified by customer orders. Each customer order is characterized by a certain due date until which all items included in the order are to be retrieved. For the actual picking process, customer orders may be grouped (batched) into more substantial picking orders (batches). The items of a batch are then collected on a picker tour. Thus, the picking process of each customer order in the batch is completed when the picker returns to the depot after the last item of the batch has been picked. Whether and to what extent due dates are violated depends on how the customer orders are batched, how the batches are assigned to order pickers, how the assigned batches are sequenced and how the pickers are routed. Existing literature has only dealt with specific aspects of this problem so far. In this paper, for the first time, an approach is proposed which considers all subproblems simultaneously. A mathematical model of the problem is introduced that allows for solving small problem instances. For larger instances, a variable neighborhood descent algorithm is presented. By means of numerical experiments, it is demonstrated that the algorithm provides solutions of excellent quality. Furthermore, it is shown that a simultaneous solution approach to the above-mentioned subproblems can be considered as a significant source for improving the efficiency of operations in distribution warehouses.

1. Introduction

Order picking is a function which is critical for managing and operating distribution warehouses efficiently. It deals with the retrieval of items requested by external or internal customers (Petersen & Schmenner, 1999; Wäscher, 2004). In picker-to-part systems, which are referred to in this paper, human operators (order pickers) walk or ride through the warehouse and collect the requested items from their storage locations (Wäscher, 2004).

The items specified by a customer order usually have to be provided until a certain due date (Henn & Schmid, 2013). Violation of due dates may delay subsequent shipment and/or production processes, and, as a consequence, results in an unacceptable customer satisfaction and high costs. Whether or to what extent due dates of a set of customer orders can be met is dependent on (1) how the customer orders are grouped into picking orders (Order Batching Problem), (2) how the picking orders are assigned to and sequenced by the order pickers (Batch Assignment and Sequencing Problem), and (3) how each order picker is routed in order to collect the items of each picking order (Picker Routing Problem). These problems are closely interrelated. Thus, solving them simultaneously appears to be a promising approach for the provision of solutions which comply with the given due dates in the best possible way. Literature dealing with solution approaches which explicitly take into account these problems simultaneously is almost non-existing. To the best of our knowledge, Chen, Cheng, Chen, and Chan (2015) represent the only exception. Their approach is related to a problem environment which is more specific than the one considered in this paper. Furthermore, computing times become a critical issue and the numerical experiments demonstrate that this approach can only be applied to very small problem instances. Due to this reason, the approach of Chen et al. (2015) is neither suitable for dealing with practical-sized problem instances nor can it be used for evaluating the benefit which results from solving the subproblems simultaneously.

* Corresponding author.
E-mail address: andre.scholz@ovgu.de (A. Scholz).

http://dx.doi.org/10.1016/j.ejor.2017.04.038
0377-2217/

Consequently, in this paper, we present a new, more competitive approach to what is called hereafter the Joint Order Batching, Assignment and Sequencing, and Routing Problem (JOBASRP) and which includes an integrative view of the problems sketched above. We propose a mathematical programing formulation to this problem whose size increases polynomially with the number of customer orders. This model provides insights into the problem but is only appropriate for solving small problem instances. Therefore, we also introduce a heuristic solution approach, namely a variable neighborhood descent algorithm, which incorporates neighborhood structures regarding the batching and the sequencing problem proposed in an earlier paper by Henn (2015). The arising routing problems are solved by means of the combined heuristic, which constructs routes of good quality within fractions of a second (Roodbergen & de Koster, 2001a). In order to improve the routes, the Lin–Kernighan–Helsgaun heuristic (Helsgaun, 2000) is applied to very promising solutions. By means of numerical experiments, it is shown that this approach leads to high-quality solutions within reasonable computing times even for large problem instances. Furthermore, a sequential approach is constructed which is composed of state-of-the-art algorithms for the respective subproblems. In the experiments, the sequential and the joint approach are compared with respect to the solution quality in order to provide an insight into the benefits of dealing with the JOBASRP as a holistic problem. It is pointed out that the joint consideration of the subproblems results in a substantial reduction of the tardiness of all customer orders and that the application of an integrated approach is inevitable in order to obtain high-quality solutions.

The remainder of this paper is organized as follows: in Section 2, we give a precise statement of the JOBASRP. Section 3 comprises a literature review regarding the subproblems and joint problems. For the JOBASRP, a new mathematical model formulation is presented in Section 4. Section 5 contains the description of the variable neighborhood descent algorithm including the generation of an initial solution, the different neighborhood structures and the integration of the routing algorithms. In Section 6, the numerical experiments are presented which have been carried out in order to evaluate the performance of the algorithm as well as the benefits resulting from solving the subproblems simultaneously. The paper concludes with an outlook on further research.

2. Problem description

We consider a warehouse with a manual, low-level picker-to-parts order picking system from which a given set of items has to be retrieved. The items are stored on pallets or in bins directly accessible to the order pickers (Henn, Koch, & Wäscher, 2012). The storage locations of the items typically constitute a block layout (Roodbergen, 2001) composed of so-called picking aisles and cross aisles. The picking aisles run parallel to each other and include storage locations arranged on both sides of each picking aisle. Cross aisles do not contain any storage locations but enable order pickers to enter or exit a picking aisle. Furthermore, the cross aisles divide the picking area into several blocks and the picking aisles into subaisles. A block is formed by the picking area located between two adjacent cross aisles. The corresponding part of a picking aisle is denoted as a subaisle. Thus, a warehouse with m picking aisles and $q+1$ cross aisles includes q blocks and $q \cdot m$ subaisles. Additionally, the warehouse contains a depot where the order pickers enter the picking area and return to in order to deposit the picked items. In Fig. 1, an example of a picking area with two blocks and five picking aisles is depicted. A two-block layout is characterized by three cross aisles, namely the front, middle and rear cross aisle, where the front and the rear cross aisles represent the cross aisles nearest to and farthest away from the depot, respectively. The middle cross aisle separates the two blocks from each other. The storage locations are represented by rectangles, while black rectangles symbolize the locations of requested items (pick locations).

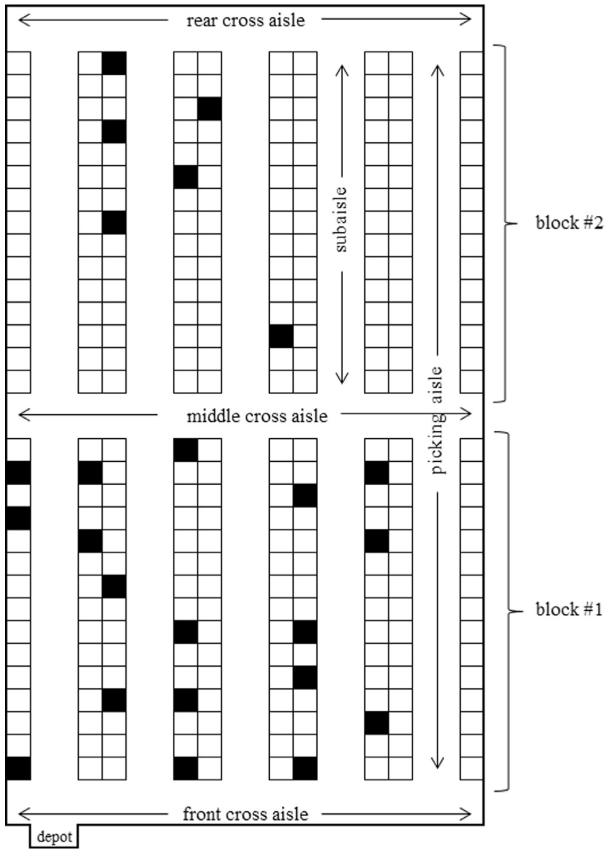


Fig. 1. Two-block layout.

In order to retrieve the requested items, several order pickers operate in the picking area. Each order picker is equipped with a picking device, e.g. a cart or a roll cage, enabling him to perform a tour through the warehouse on which several items are picked before he returns to the depot. The maximum number of stops on a tour is dependent on the capacity of the picking device and the capacity requirements of the respective customer orders. On his tour, the order picker is guided by a so-called pick list. The list represents a batch and identifies the storage locations and the quantities of the items which are to be retrieved on the same tour. A batch may include requested items of several customer orders. However, splitting of customer orders is not allowed since it would result in an unacceptable sorting effort. The pick list also contains information on the sequence according to which a picker is meant to visit the respective pick locations.

The time an order picker spends for retrieving all items of a batch (batch processing time) can be divided into (Tompkins, White, Bozer, & Tanchoco, 2010):

- the setup time, i.e. the time needed for preparing a tour,
- the search time, i.e. the time required at each pick location for identifying the correct item,
- the pick time, i.e. the time for physically retrieving the items from their storage locations, and

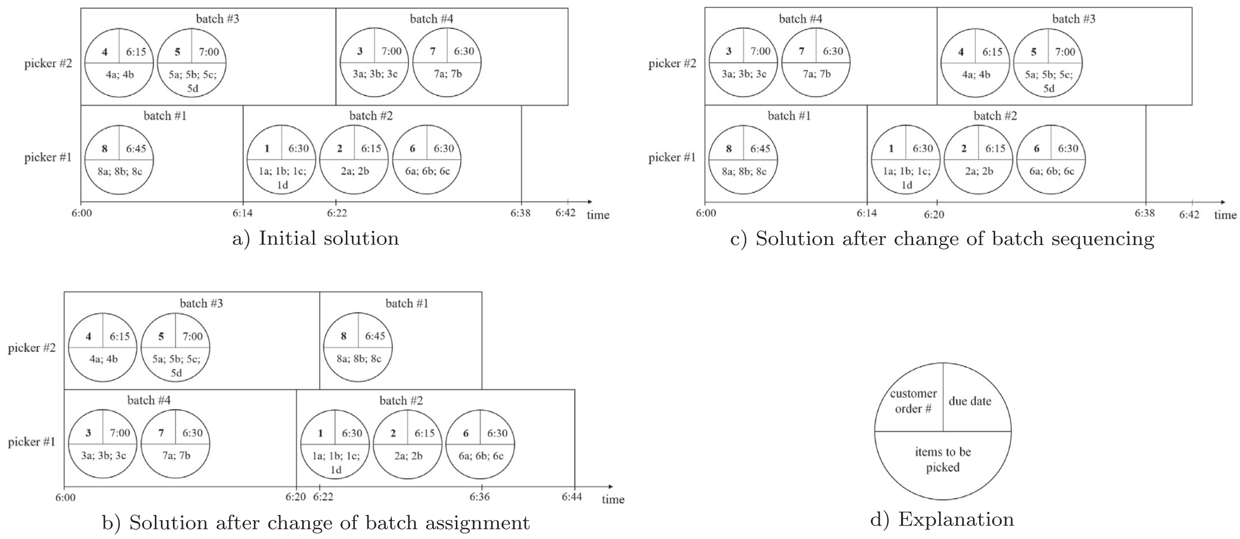


Fig. 2. Impact of batch assignment and sequencing decisions.

- the travel time, i.e. the time spent for traveling from the depot to the first pick location, between the pick locations and from the last pick location back to the depot.

Processing of a batch is started when the corresponding order picker to whom the batch has been assigned starts with preparing the tour. This point in time is addressed as the batch start time, while the batch completion time denotes the point in time when the picker returns to the depot after having retrieved all items included in the batch. The start time (likewise: the completion time) of a customer order, therefore, is identical to the start time (completion time) of the batch in which the order is included. An example including 8 customer orders to be processed by two order pickers is given in Fig. 2. In Fig. 2(a), a Gantt chart is depicted in which each batch is represented by a box. Each batch consists of several customer orders, each of which, again, including several items to be picked, e.g. batch #4 includes order #3 (items to be picked: 3a, 3b, 3c) and order #7 (items to be picked: 7a, 7b). According to this chart (and based on the given definition) the completion times of customer orders #3 and #7 are identical and both orders are completed at 6:42.

Given a particular composition of the batches, a different assignment of batches to order pickers (see Fig. 2(b)) and/or a different sequence of the batches assigned to a picker (see Fig. 2(c)) will result in different order completion times. From Fig. 2(b) and (c) we take, e.g. that the completion time of customer orders #3 and #7 now is 6:20, respectively. Of course, according to Fig. 2(b), also the completion times of the orders included in batch #1 (order #8), in batch #2 (orders #1, #2 and #6), and according to Fig. 2(c), also the completion time of batch #3 (orders #4 and #5) has been affected.

Furthermore, the completion time of customer orders can also be influenced by the assignment of orders to batches. The length of a box, i.e. its extent in the horizontal dimension, indicates the batch processing time. It is determined by the customer orders from which it is composed and by the time the order picker needs to complete a respective tour through the warehouse on which the requested items are collected. In Fig. 3(a), the schedule of Fig. 2(a) is repeated. An example of a tour corresponding to batch #2 is presented in Fig. 3(b). In Fig. 3(c), the batches are composed differently. Batch #2 now includes orders #2, #6 and #7, while order #1 has been assigned to batch #4. The corresponding picker tour through the warehouse related to batch #2, depicted in Fig. 3(d), is different to the one of Fig. 3(b). In particular, since the tour is now shorter, less time will be necessary for its completion, resulting in a decrease of the processing time of batch #2, i.e. a decrease of the length of the box related to batch #2 in Fig. 3(c). Likewise, also the processing time of batch #4 to which order #1 has been assigned will be affected. We note again that also completion times of all customer orders may be affected which are included in batches scheduled after those batches to which such changes occurred.

In distribution warehouses, customer orders usually have to be picked until certain due dates (Henn & Schmid, 2013) in order to guarantee the scheduled departure of trucks delivering the picked items during contracted (and often tight) time windows to external customers (Gademann, van den Berg, & van der Hoff, 2001). As can be taken from the examples presented in Figs. 2 and 3, whether such due dates are met or not depends on the assignment of customer orders to batches, the assignment of batches to order pickers, the sequencing of batches and the routing of order pickers. In the example from Fig. 2, orders #7 and #8 are due at 6:30 and 6:45, respectively. According to the solution depicted in Fig. 2(a), order #7 is completed at 6:42, i.e. it is delayed by 12 minutes, whereas order #8 is completed in time. In Fig. 2(b), the assignment of the batches, which include the two orders, to the order pickers is changed. Order #7 is now completed at 6:20, while the completion time of order #8 is 6:36. Thus, the due dates of both orders are met. However, this affects the completion time of batch #2 which is increased by one minute and, therefore, the extent to which orders #1, #2 and #6 are delayed is also increased.

Delayed shipments result in fines which, e.g. for the delivery of fresh foods to supermarkets or of parts to production lines in the car industry, are often dependent on the length of the delay (tardiness). More precisely, the tardiness τ_n of a customer order n is defined as the (non-negative) difference between the completion time c_n of the order and its due date d_n (Henn & Schmid, 2013), i.e. the tardiness is given by $\tau_n = \max\{c_n - d_n; 0\}$. The sum of the tardiness of all customer orders is referred to as the total tardiness. A total tardiness of 0 means that the due dates of all customer orders are met in the respective solution. The total tardiness of all customer orders will be used here for the evaluation of solutions provided by our approach. We note that this objective

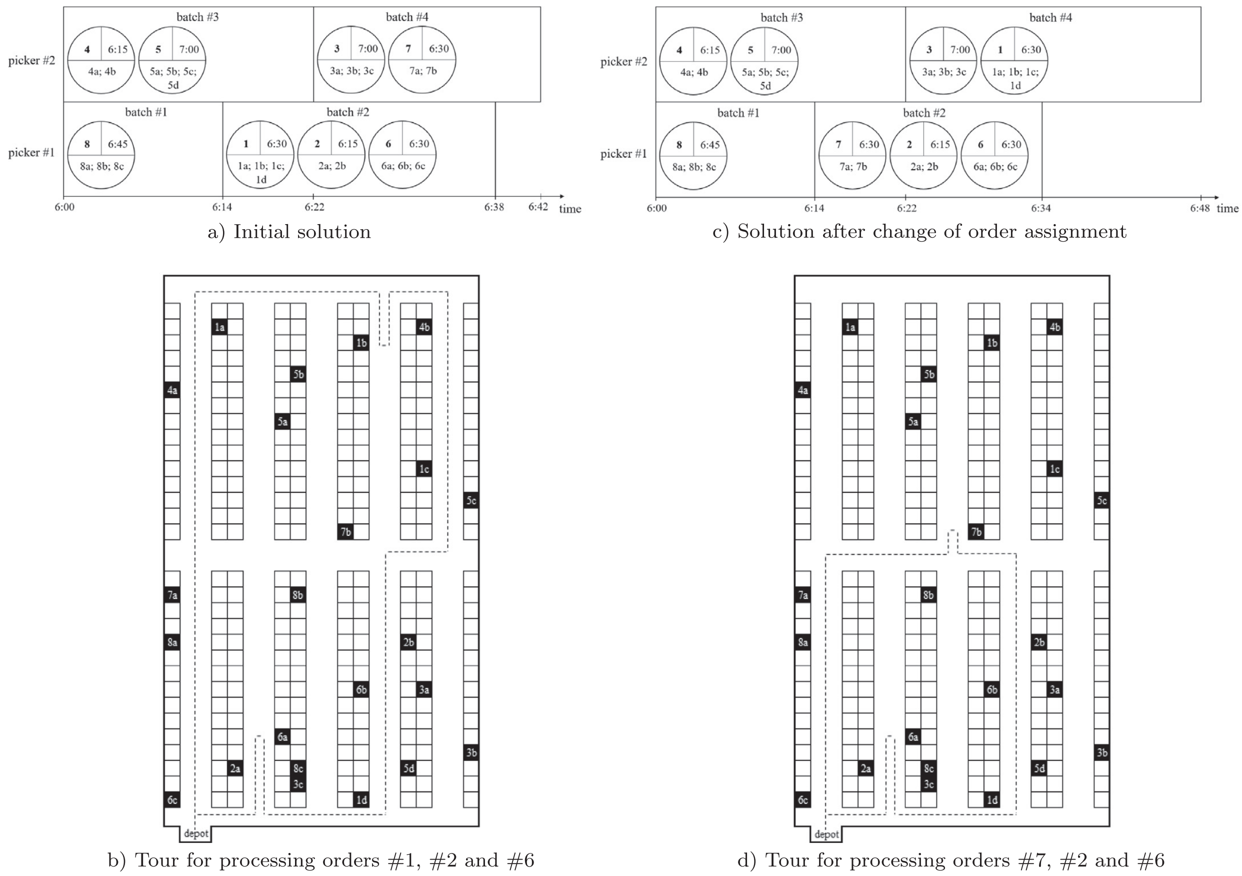


Fig. 3. Impact of order batching decisions on picker routing.

is widely used; not only in the literature related to order picking (Elsayed, Lee, Kim, & Scherer, 1993; Henn & Schmid, 2013; Tsai, Liou, & Huang, 2008) but also in literature related to other problem settings in which on-time delivery to external or internal clients represents a key element of the study, e.g. in scheduling (Koulamas, 1994; Pinedo, 2016; Ullrich, 2013).

The Joint Order Batching, Assignment and Sequencing, and Routing Problem can now be stated as follows: let a non-empty set of customer orders be given, each of which including certain items with known storage locations to be removed from the warehouse. Furthermore, each order is characterized by a due date until which all requested items of the order should be retrieved and brought forward to the depot. A given number of order pickers is available for carrying out the necessary picking operations. Then, the following questions have to be answered (simultaneously) in such a way that the total tardiness is minimized:

- How should the set of customer orders be grouped into picking orders? (Order Batching Problem)
- How and in which sequence should the set of picking orders be assigned to the order pickers? (Batch Assignment and Sequencing Problem)
- For each picking order, in which sequence should the respective pick locations be visited? (Picker Routing Problem)

In the following, we will assume a warehouse with wide aisles which enable the order pickers to pass each other. This assumption has also been made by Chen et al. (2015) and Henn (2015) and allows for neglecting cases in which the processing time of a batch is increased by waiting times which may arise if order pickers simultaneously work in the same picking aisle. Even without considering such picker blocking aspects, the JOBASRP formulated above is known to be NP-hard (Chen et al., 2015).

3. Literature review

Although the above-mentioned subproblems of the JOBASRP arise simultaneously, joint solution approaches have rarely been addressed in the literature so far. Instead, the subproblems are dealt with independently of each other in most approaches. The first subproblem is the Order Batching Problem (OBP) which can be stated as follows (Wäscher, 2004): given the article storage locations, the routing strategy to be used, and the capacity of the picking device, how can the set of customer orders be grouped into picking orders such that the total lengths of the arising tours is minimized? The OBP has been widely studied in the literature and a large variety of solution approaches exists. For a very detailed review of solution methods to the OBP, we refer to de Koster, Le-Duc, and Roodbergen (2007) and Henn et al. (2012).

As for the Batch Assignment and Sequencing Problem (BASP) customer orders are characterized by due dates which have to be met in the best possible way, while each customer order has already been assigned to a certain batch. Solutions are evaluated by the total tardiness of all orders. The BASP can then be stated as follows: how should the batches be assigned to a limited number of pickers and, for each picker, how should the batches be sequenced such that the total

Table 1
Joint considerations regarding the Batching, Assignment and Sequencing, and Routing Problem.

Reference	Batching	Assignment	Sequencing	Routing
Kulak, Sahin, and Taner (2012)	✓			✓
Grosse, Glock, and Ballester-Ripoll (2014)	✓			✓
Scholz and Wäscher (2017)	✓			✓
Elsayed et al. (1993)	✓		✓	
Elsayed and Lee (1996)	✓		✓	
Henn and Schmid (2013)	✓		✓	
Henn (2015)	✓	✓	✓	
Tsai et al. (2008)	✓		✓	✓
Chen et al. (2015)	✓		✓	✓

tardiness is minimized? The special cases of the BASP with a single customer order per batch or with customer orders having identical due dates find their parallels in Parallel Machine Scheduling: a set of jobs (here: batches) has to be assigned to machines (here: order pickers) and sequenced in such a way that the total tardiness of all jobs is minimized (Pinedo, 2016). We refer to Koulamas (1994) for a review of solution approaches to the Parallel Machine Scheduling Problem. To the best of our knowledge, no solution approach exists to the BASP where the batches include customer orders with different due dates.

The Picker Routing Problem (PRP) can be stated as follows (Ratliff & Rosenthal, 1983; Scholz, Henn, Stuhlmann, & Wäscher, 2016): given a set of items to be picked from known storage locations, in which sequence should the locations be visited such that the total length of the corresponding picker tour is minimized? Although the PRP can be solved efficiently in warehouses with up to two blocks (Ratliff & Rosenthal, 1983; Roodbergen & de Koster, 2001b), simple routing strategies are used for routing order pickers in practice. This is due to the fact that optimal tours appear to be complex and difficult to memorize for the order pickers. The S-shape strategy is the most frequently used routing scheme in practice (Roodbergen, 2001). When applying this strategy, the order picker traverses each subaisle completely which contains at least one requested item. An exception may occur in the last subaisle of the block which is visited. Here, the picker may return after retrieving all items in this subaisle. A more sophisticated strategy, the so-called combined strategy (Roodbergen & de Koster, 2001a), combines elements of the S-shape and the return strategy. For each picking aisle to be visited, by means of dynamic programing it is determined whether the aisle is traversed or whether it is entered and left via the same cross aisle. For a detailed review of routing strategies, we refer to de Koster et al. (2007) (single-block layout) and Roodbergen and de Koster (2001a) (multi-block layout). As the PRP is a special case of the Traveling Salesman Problem (TSP), Theys, Bräysy, Dullaert, and Raa (2010) applied a TSP heuristic, namely the Lin–Kernighan–Helsgaun heuristic (Helsgaun, 2000), to the PRP and demonstrated that the tour length can be reduced by up to 48% compared to the usage of the simple S-shape strategy.

In more recent years, research has shifted to solution approaches which focus on the simultaneous solution of two or more subproblems of the JOBASRP. Table 1 provides an overview of respective publications. The table shows that all approaches deal with the OBP and either include the BASP or the PRP. As for the BASP, the assignment and the sequencing problem are treated separately. A check mark provided in the sequencing but not in the assignment column indicates that a single picker is considered only, i.e. no assignment decisions have to be made in this case. In the following part of the review, we focus on those articles dealing with three of the four decision types.

In Henn (2015), the OBP is combined with the BASP, resulting in the Joint Order Batching, Assignment and Sequencing Problem (JOBASP). This is the only paper so far in which multiple pickers have been taken into account. Henn (2015) proposed a variable neighborhood descent (VND) and a variable neighborhood search (VNS) approach in order to solve the JOBASP. As an upper bound, an earliest start date rule (ESDR)-based algorithm has been chosen in which customer orders are batched and sequenced according to their due dates, and batches are assigned to the picker who currently possesses the smallest total processing time. Problem instances with up to 200 customer orders have been included in the numerical experiments. The total tardiness obtained by using the ESDR-based algorithm could be reduced by 41% (VNS) and 39% (VND) on average. Application of the VNS algorithm requires up to 25 minutes of computing time, while the VND approach terminates after a maximum of 30 seconds.

In general, from all the papers discussed so far, it becomes clear that the simultaneous solution of the subproblems of the JOBASRP may provide large benefits w.r.t. improved planning of picking operations. Nevertheless, only two approaches exist which deal with the complete JOBASRP. Tsai et al. (2008) proposed a genetic algorithm for the JOBASRP. Apart from the total tardiness, they also minimize the total earliness as well as the total tour length. Unlike in the approaches discussed before, splitting of customer orders is allowed. Chen et al. (2015) presented a genetic algorithm for the JOBASP and solved the arising PRPs by means of an ant colony approach. In the genetic algorithm, however, the batching and the sequencing problem are considered separately which results in many unfeasible solutions with respect to the capacity constraint of the picking device. Furthermore, the ant colony algorithm consumes far more computing time than problem-specific approaches to the PRP do. Thus, it is not surprising that the authors stated that computing times are a critical issue (without reporting any computing times, though). In their numerical experiments, only very small problem instances with up to 8 orders have been considered. Both the study of Tsai et al. (2008) and Chen et al. (2015) have in common that it is assumed that only a single picker is available for processing the customer orders. This implies that the assignment problem does not have to be taken into account which simplifies the problem considerably.

We conclude that, even though the joint solution of the subproblems of the JOBASRP seems to represent a promising research path, no approach exists which captures the core elements of the problem and, at the same time, is capable of providing solutions of good quality in reasonable computing time.

4. Model formulation

The mathematical model introduced by Henn (2015) for the JOBASP can be adapted to the JOBASRP. It requires that all feasible batches have to be generated in advance. Then, for each feasible batch, the minimum processing time has to be computed, which involves solving the arising PRPs to optimality. As a consequence, when solving a specific problem instance, providing the problem data for the model would already consume a large amount of computing time. Furthermore, the number of variables in the

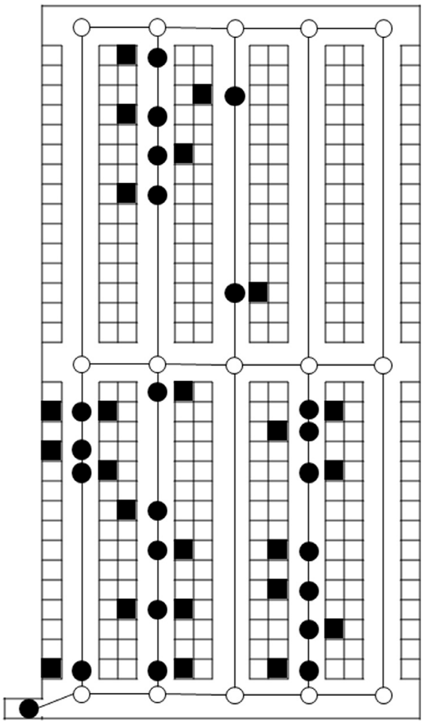


Fig. 4. Illustration of a Steiner TSP.

model depends on the number of feasible batches, which increases exponentially with the number of customer orders. Thus, even for small instances, it may not be possible to generate the model explicitly at all due to memory restrictions.

Instead, in order to keep the number of variables at a reasonable level, we chose a modeling approach in which the batches are not generated in advance. For our model formulation as well as for the heuristic approach presented in Section 5, we assume that the capacity of the picking device is measured in the number of items, i.e. a maximum number of C items can be picked on a tour. This is a standard assumption which has also been made by Bozer and Kile (2008) and Henn (2015). Note, that this is not a critical assumption as the mathematical model and the heuristic approach can easily be modified such that other kinds of capacity constraints (e.g. a maximum number of customer orders or a maximum total weight of items) can be dealt with.

The model formulation can be divided into two parts: the first part is related to the Joint Order Batching and Picker Routing Problem which means that the customer orders are grouped into batches and the corresponding tours are constructed. With respect to the specific structure of tours in an order picking warehouse, we interpret the routing problem as a Steiner TSP. The graph of Fig. 4 illustrates this interpretation and will be used as the basis for our model.

Black vertices represent the location of the depot and the pick locations. These vertices have to be included in the tour. White vertices (Steiner points) depict the intersections of picking aisles and cross aisles. They can be visited, but do not necessarily need to be included in the tour. As can be taken from the figure, the maximum degree of a vertex is equal to four. Therefore, this Steiner TSP-based graph contains significantly fewer arcs than a standard TSP graph would include. In the model formulation, we consider a directed graph which means that each edge is replaced by two reverse arcs.

The second part of the model formulation deals with the BASP in which the constructed batches are assigned to a picker and arranged in a certain sequence. The sequence consists of as many positions as batches have been assigned to the picker. The picker starts processing the batch assigned to the first position and, after having returned to the depot, the batch in the next position is dealt with etc. This procedure provides the completion time for each batch and the completion times of all customer orders included in the batch, from which the tardiness of each customer order can be determined.

Before presenting the model formulation, we introduce the corresponding sets, parameters and variables.

Sets

- P: set of order pickers
- K: set of positions (for each order picker) where a batch can be scheduled $\left(K=\left\{1,\ldots,\overline{K}\right\}\right)$
- H: set of tours which can be performed
- N: set of customer orders
- V: set of vertices in the graph representing the warehouse
- R: set of vertices representing a pick location and the location of the depot
- A: set of arcs in the graph representing the warehouse

Parameters

- C: capacity of the picking device
- β^s: setup time per batch
- β^p: pick and search time per item
- β^t: travel time per length unit
- c_n: number of requested items of order $n \in N$
- d_n: due date of order $n \in N$
- α_{nr}: constant for indicating whether vertex $r \in R$ represents a storage location of a requested item of order $n \in N$ $(\alpha_{nr}=1)$ or not $(\alpha_{nr}=0)$
- w_a: distance to be covered when using arc $a \in A$
- M: sufficiently large number

$$\left(\text{e.g. } M=\overline{K}\cdot\left(\beta^s+\beta^t\cdot\sum_{a\in A}w_a\right)+\beta^p\cdot\sum_{n\in N}c_n\right)$$

Variables

- τ_n: tardiness of order $n \in N$
- $\tilde{u}_h$: processing time of tour $h \in H$
- u_{pk}: processing time of the tour assigned to position $k \in K$ of picker $p \in P$
- v_{pk}: completion time of the tour assigned to position $k \in K$ of picker $p \in P$
- x_{pkh}: variable for indicating whether tour $h \in H$ is assigned to position $k \in K$ of picker $p \in P$ $(x_{pkh}=1)$ or not $(x_{pkh}=0)$
- y_{nh}: variable for indicating whether order $n \in N$ is assigned to tour $h \in H$ $(y_{nh}=1)$ or not $(y_{nh}=0)$
- z_{ah}: variable for indicating whether arc $a \in A$ is included in tour $h \in H$ $(z_{ah}=1)$ or not $(z_{ah}=0)$
- f_{ah}: number of units of the commodity passing arc $a \in A$ on tour $h \in H$

The JOBASRP can then be formulated as follows.

$$\min \sum_{n\in N}\tau_n \tag{1}$$

$$\sum_{h\in H}x_{pkh}\leq 1 \qquad \forall\, p\in P,\ k\in K \tag{2}$$

$$\sum_{p\in P}\sum_{k\in K}x_{pkh}=1 \qquad \forall\, h\in H \tag{3}$$

$$\sum_{h \in H} y_{nh} = 1 \qquad \forall n \in N \tag{4}$$

$$\sum_{n \in N} c_n \cdot y_{nh} \le C \qquad \forall h \in H \tag{5}$$

$$\sum_{a \in \delta_0^-} z_{ah} \ge 1 \qquad \forall h \in H \tag{6}$$

$$\sum_{a \in \delta_r^-} z_{ah} \ge \alpha_{nr} \cdot y_{nh} \qquad \forall h \in H,\ n \in N,\ r \in R \backslash \{0\} \tag{7}$$

$$\sum_{a \in \delta_v^-} z_{ah} = \sum_{a \in \delta_v^+} z_{ah} \qquad \forall h \in H,\ v \in V \tag{8}$$

$$\sum_{a \in \delta_r^+} f_{ah} - \sum_{a \in \delta_r^-} f_{ah} = \alpha_{nr} \cdot y_{nh} \qquad \forall h \in H,\ n \in N,\ r \in R \backslash \{0\} \tag{9}$$

$$\sum_{a \in \delta_v^+} f_{ah} - \sum_{a \in \delta_v^-} f_{ah} = 0 \qquad \forall h \in H,\ v \in V \backslash R \tag{10}$$

$$f_{ah} \le C \cdot z_{ah} \qquad \forall a \in A,\ h \in H \tag{11}$$

$$\beta^s + \beta^p \cdot \sum_{n \in N} c_n \cdot y_{nh} + \beta^t \cdot \sum_{a \in A} w_a \cdot z_{ah} \le \tilde{u}_h \qquad \forall h \in H \tag{12}$$

$$\tilde{u}_h - M \cdot \left(1 - x_{pkh}\right) \le u_{pk} \qquad \forall p \in P,\ k \in K,\ h \in H \tag{13}$$

$$u_{p1} \le v_{p1} \qquad \forall p \in P \tag{14}$$

$$u_{pk} + v_{p,k-1} \le v_{pk} \qquad \forall p \in P,\ k \in K \backslash \{1\} \tag{15}$$

$$v_{pk} - d_n - M \cdot \left(2 - x_{pkh} - y_{nh}\right) \le t_n \qquad \forall n \in N,\ p \in P,\ k \in K,\ h \in H \tag{16}$$

$$x_{pkh},\ y_{nh},\ z_{ah} \in \{0, 1\} \qquad \forall a \in A,\ n \in N,\ p \in P,\ k \in K,\ h \in H \tag{17}$$

$$\tau_n,\ \tilde{u}_h,\ u_{pk},\ v_{pk},\ f_{ah} \ge 0 \qquad \forall a \in A,\ n \in N,\ p \in P,\ k \in K,\ h \in H \tag{18}$$

The objective function (1) minimizes the total tardiness. Constraints (2) ensure that at most one tour is assigned to each position of each picker, while (3) guarantee that each tour is performed. Each customer order has to be processed in exactly one tour which is obtained by meeting restrictions (4). The capacity of the picking device is taken into account by satisfying (5). Constraints (6)–(11) represent the routing constraints. First, constraints (6) ensure that the depot is left on each tour. Here, δ_v^+ (δ_v^-) denotes the set of arcs to which vertex v is an end (start) vertex. Vertex "0" represents the location of the depot. Constraints (7) guarantee that each pick location is visited which corresponds to a requested item of a customer order included in the respective tour. Restrictions (8) are the degree constraints. The following three types of constraints represent subtour elimination constraints. These constraints are related to the single-commodity flow constraints introduced by Letchford, Nasiri, and Theis (2013). Subtours are excluded by ensuring that the picker starts his tour with a certain number of units of a commodity and delivers one unit to each vertex to be visited. In this way, the vertices are enumerated according to their appearance in the tour. In constraints (12), the processing time is determined for each tour, which is composed of the setup time, the time for searching and picking the items and the travel time. The processing time of the tour assigned to a certain position of a certain picker is determined in (13), while constraints (14) and (15) calculate the corresponding completion times. Finally, constraints (16) compute the tardiness for each order. The variable domains are defined in (17) and (18).

The model formulation includes linear constraints only. Furthermore, both the number of variables and the number of constraints increase polynomially with the problem size, which is a major advantage of this model in comparison to the formulations of Chen et al. (2015) and Henn (2015). However, as will be shown later, our model is only suitable for solving relatively small problem instances. For dealing with larger instances, we have developed a variable neighborhood descent approach to the JOBASRP.

5. Variable neighborhood descent

5.1. Overview

Variable neighborhood descent was introduced in Hansen and Mladenović (2001). The general principle of VND consists of exploring the solution space of the problem by means of a sequence of neighborhood structures $\mathcal{N}_1, \ldots, \mathcal{N}_L$. It is started with an incumbent solution s^* and the best neighbor s (in terms of the objective function value) of $\mathcal{N}_1(s^*)$ is determined. If s represents a better solution than s^*, then s becomes the new incumbent solution and the first neighborhood structure $\mathcal{N}_1(s^*)$ is considered again. Otherwise, the exploration of the solution space continues with the next neighborhood. The algorithm terminates if no improvement can be found in the last neighborhood structure $\mathcal{N}_L(s^*)$, i.e. a local optimum is found with respect to all neighborhood structures.

In our VND approach, a solution s is a solution to the JOBASP, i.e. it includes information about how the orders are grouped into batches and in which sequence the batches are to be processed by the pickers. Based on this solution, different PRPs have to be solved in order to determine the corresponding objective function values. Since many PRPs arise during the solution process, we decided to solve them by applying the combined heuristic. This heuristic was particularly designed for warehouses with multiple blocks and outperforms the frequently used S-shape heuristic by far in terms of solution quality (Roodbergen & de Koster, 2001a). The corresponding objective function value is denoted by $f_{\text{Comb}}(s)$.

In contrast to the standard VND procedure, we do not directly return to $\mathcal{N}_1$ after an improvement has been found. Instead, before doing so, we determine a local optimum regarding the current neighborhood structure. The objective function value, the total tardiness, is strongly dependent on the processing times of the batches. In order to reduce processing times, each time when a local optimum has been found, the arising PRPs are solved by means of the Lin–Kernighan–Helsgaun (LKH) heuristic. Applied to the PRP, this heuristic results in very short tours (Theys et al., 2010) and, as a consequence, in shorter processing times. However, since the computational effort is much higher than the effort for the application of the combined heuristic, it is not possible to use the LKH heuristic for evaluating all solutions to be considered in the exploration of the neighborhood structures. Due to this fact, we try to ensure that the LKH heuristic is only applied to very promising solutions. Therefore, we decided to determine a local optimum before returning to $\mathcal{N}_1$. $f_{\text{LKH}}(s)$ denotes the total tardiness of a solution s whose PRPs have been solved by means of the LKH heuristic. A pseudocode of our VND approach is depicted below, while the generation of the initial solution as well as the neighborhood structures will be dealt with in the next sections.

5.2. Initial solution

For the generation of an initial solution, two constructive approaches are first applied, providing one solution each. The solution with the smaller objective function value is then taken as the initial solution for the VND. The first constructive approach is based on the earliest start date rule and was also used by Henn (2015). It is a priority rule-based algorithm in which orders are assigned successively to the batches and the positions of the pickers. A detailed pseudocode of this algorithm is depicted below.

In the pseudocode, U denotes the set of orders not yet assigned to a picker, k_p is the sequence position of picker p to which the next batch can be assigned and B_{pk_p} represents the set of orders included in the batch assigned to position k_p of picker p. C_p and v_p denote the number of items contained in the batch under consideration for picker p as well as its completion time, while u_{Comb} calculates the processing time of the current batch by means of the combined heuristic. At the beginning of the algorithm, all orders are unassigned. The orders are then assigned successively to certain batches, starting with the unassigned order n^* with the smallest due date. For each order picker p, the completion time $\tilde{v}_p$ is determined that would follow from an assignment of n^* to the picker. The assignment consists of an addition of n^* to batch B_{pk_p} if the capacity constraint of the picking device is not violated; otherwise, the assignment includes the opening of a new batch containing n^*. The algorithm terminates when all orders have been assigned to batches and positions.

In order to minimize the total tardiness, the orders are sorted according to their due dates and then assigned successively. The composition of the batches is not considered any further. This is a reasonable approach as long as the due dates are loose. However, in case of tight due dates, it is of prime importance to construct batches and tours which allow for short processing times. We, therefore, propose another constructive approach which takes into account the processing times resulting from the construction of the batches. This approach can be perceived as a seed algorithm. Seed algorithms have been frequently applied to the OBP and consist of two steps (Elsayed, 1981). First, a seed order is chosen by means of a seed selection rule and assigned to a new batch. According to an order addition rule, orders are then added to this batch. As for the seed selection, we chose an order which is not yet assigned to a batch and has the closest due date. Dependent on the savings in terms of total tardiness, which result from adding an order to this batch instead of processing it separately, other orders are assigned to the batch. A detailed pseudocode of this algorithm can be seen in Algorithm 3.

In step 1, each order is assigned to a separate batch. Starting with the batch that includes the order with the smallest due date, the batches are successively added to the picker p who currently possesses the shortest completion time $\tilde{v}_p$. In step 2, batches are merged in order to reduce the processing times as well as the total tardiness. The order with the smallest due date represents the seed order and forms batch $\tilde{B}_1$. Based on the solution generated in step 1, for each unassigned order $n \in U$, savings sav_{in} are determined which are defined as the reduction of the total tardiness obtained by merging the batch of order n and the batch i of the seed order. Of the potential pairs of batches which could be merged without violating the capacity constraint, two batches are actually merged for which the savings are maximal. Then, the savings are updated and another order may be added to the batch. This is done until no further positive savings can be realized. While at least one order exists not considered in step 2, i. e. while $U \neq \emptyset$, the next seed order is determined based on the due dates and a new batch is opened to which orders are to be added. Thus, step 2 provides a solution in which the processing times and the total tardiness are explicitly taken into account. In step 3, the batches in this solution are reassigned to the positions of the pickers. This is done consecutively starting with batch $\tilde{B}_1$. Each batch is inserted into the position k of picker p minimizing the total tardiness $f^{p,k}_{\text{Comb}}$ of all batches which have been inserted up to this point. In analogy to the ESDR-based algorithm, the combined heuristic is used to determine the processing times.

Algorithm 1 Variable neighborhood descent algorithm for the JOBASRP.

Input: problem data, number of neighborhood structures L
Output: solution s^* to the JOBASRP and corresponding total tardiness $f_{\text{LKH}}(s^*)$

generate initial solution s;
$s^* := s$; $l := 1$;
while $l \le L$ **do**
  $s := s^*$;
  $s' := \arg\min\{f_{\text{Comb}}(\tilde{s}) \mid \tilde{s} \in \mathcal{N}_l(s)\}$;
  while $f_{\text{Comb}}(s') < f_{\text{Comb}}(s)$ **do**
    $s := s'$;
    $s' := \arg\min\{f_{\text{Comb}}(\tilde{s}) \mid \tilde{s} \in \mathcal{N}_l(s)\}$;
  end while
  if $f_{\text{LKH}}(s) < f_{\text{LKH}}(s^*)$ **then**
    $s^* := s$;
    $l := 1$;
  else
    $l := l + 1$;
  end if
end while

Algorithm 2 ESDR-based algorithm.

Input: set of orders N with due dates d_n and number of requested items c_n $(n \in N)$, set of pickers P, capacity C of the picking device
Output: solution s^* to the JOBASRP and corresponding total tardiness $f_{\text{Comb}}(s^*)$

$U := N$;
for $p \in P$ **do**
  $k_p := 1$; $B_{pk_p} := \emptyset$; $C_p := 0$; $v_p := 0$;
end for
while $U \neq \emptyset$ **do**
  $n^* = \arg\min\{d_n \mid n \in U\}$;
  for $p \in P$ **do**
    if $C_p + c_{n^*} \le C$ **then**
      $\tilde{v}_p := v_p + u_{\text{Comb}}\big(B_{pk_p} \cup \{n^*\}\big)$;
    else
      $\tilde{v}_p := v_p + u_{\text{Comb}}(\{n^*\})$;
    end if
  end for
  $p^* := \arg\min\{\tilde{v}_p \mid p \in P\}$;
  $U := U \setminus \{n^*\}$; $v_{p^*} := \tilde{v}_{p^*}$;
  if $C_{p^*} + c_{n^*} \le C$ **then**
    $B_{p^*k_{p^*}} := B_{p^*k_{p^*}} \cup \{n^*\}$; $C_{p^*} := C_{p^*} + c_{n^*}$;
  else
    $k_{p^*} := k_{p^*} + 1$; $B_{p^*k_p} := \{n^*\}$; $C_{p^*} := c_{n^*}$;
  end if
end while

Algorithm 3 Seed algorithm.

Input: set of orders N with due dates d_n and number of requested items c_n $(n \in N)$, set of pickers P, capacity C of the picking device

Output: solution s^* to the JOBASRP and corresponding total tardiness $f_{\text{Comb}}(s^*)$

$U := N$; //step 1
for $p \in P$ **do**
  $k_p := 1$; $\nu_p := 0$;
end for
while $U \neq \emptyset$ **do**
  $n^* := \arg\min\{d_n \mid n \in U\}$;
  for $p \in P$ **do**
    $\tilde{\nu}_p := \nu_p + u_{\text{Comb}}(\{n^*\})$;
  end for
  $p^* := \arg\min\{\tilde{\nu}_p \mid p \in P\}$;
  $U := U \setminus \{n^*\}$; $\nu_{p^*} := \tilde{\nu}_{p^*}$; $k_{p^*} := k_{p^*} + 1$; $B_{p^* k_p} := \{n^*\}$;
end while
$U := N$; $i := 1$; //step 2
while $U \neq \emptyset$ **do**
  $n^* := \arg\min\{d_n \mid n \in U\}$; $\tilde{B}_i := \{n^*\}$; $\tilde{C} := c_{n^*}$;
  $U := U \setminus \{n^*\}$;
  while $\max\left\{sav_{in} \mid n \in U : \tilde{C} + c_n \leq C\right\} > 0$ **do**
    $n^* := \arg\max\left\{sav_{in} \mid n \in U : \tilde{C} + c_n \leq C\right\}$;
    $\tilde{B}_i := B_i \cup \{n^*\}$; $\tilde{C} := \tilde{C} + c_n$; $U := U \setminus \{n^*\}$;
  end while
  $i := i + 1$; $\tilde{C} := 0$;
end while
for $p \in P$ **do** //step 3
  $k_p := 1$;
end for
for $j := 1$ to i **do**
  $(p^*, k^*) := \arg\min\left\{f^{p,k}_{\text{Comb}} \mid p \in P, k \in \{1, \ldots k_p\}\right\}$;
  $B_{p^*,k^*+1} := B_{p^*,k^*}$; …; $B_{p^*,k_p+1} := B_{p^*,k^p}$; $B_{p^*,k^*} := \tilde{B}_i$;
  $k_{p^*} := k_{p^*} + 1$;
end for

Since the seed algorithm also considers the composition of the batches instead of just basing the construction of a solution on the information about the due dates, it can be expected to lead to better results in case of tight due dates. By selecting the best solution of the ESDR-based and the seed algorithm, we aim to generate an initial solution enabling the VND to proceed faster to a local optimum.

5.3. Neighborhood structures

As mentioned before, a solution to the JOBASRP includes information about the composition of the batches and their assignment to the positions of the pickers. The arising routing problems are only solved in order to determine the resulting objective function value. Thus, the neighborhood structures considered in our VND can be divided into structures related to the sequencing problem and structures regarding the batching problem.

Neighborhood structures related to the sequencing problem only deal with the assignment of the batches to the pickers' positions. The composition of the batches remains unchanged. Thus, complete batches are considered instead of single customer orders. Since the number of batches is usually much smaller than the number of customer orders, these neighborhoods can be explored in reasonable computing time. Therefore, these neighborhoods are used at the beginning of the VND algorithm. As proposed by Henn (2015), we use one sequencing related neighborhood structure $\mathcal{N}_1$.

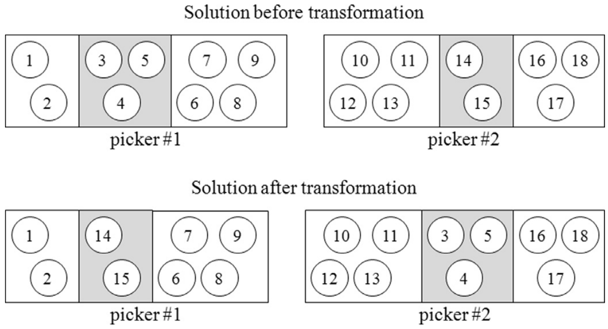


Fig. 5. Example of neighborhood structure $\mathcal{N}_1$.

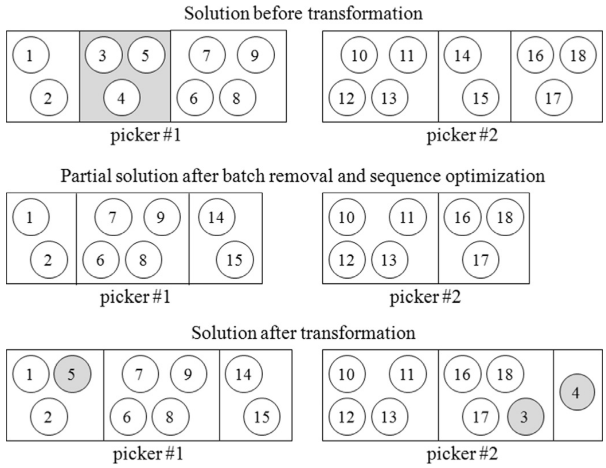


Fig. 6. Example of neighborhood structure $\mathcal{N}_2$.

Regarding $\mathcal{N}_1$, the neighborhood of a solution s is composed of all solutions which can be obtained by exchanging two batches b_1 and b_2 from s. Only exchanges between different pickers are taken into consideration. Batch b_1 is moved to the position of b_2 and vice versa. The positions of the remaining batches are not changed (see Fig. 5).

In $\mathcal{N}_2$, we consider complete batches, too. However, not only the way how batches are sequenced is changed but also how they are composed of orders. This neighborhood structure is meant to break up a complete batch and reassign the orders to other batches. Furthermore, the batch sequence is optimized. A neighbor regarding $\mathcal{N}_2$ is obtained as follows: in a first step, a batch is removed from the solution. In a second step, the remaining sequence is optimized. First, for each batch, it is checked whether the solution can be improved by moving this batch to a position of another picker. The move which provides the largest improvement is carried out. This procedure is repeated until no further improvement is possible. Movements regarding $\mathcal{N}_1$ are then applied as long as the solution can be improved. Subsequent to this improvement step, the orders contained in the removed batch are considered successively in the order of non-descending due dates. Each order may either be inserted in an existing batch or forms a new batch. The option resulting in the smallest total tardiness is realized. An example of a move regarding $\mathcal{N}_2$ is depicted in Fig. 6.

Neighborhood structures $\mathcal{N}_3$ to $\mathcal{N}_6$ are straightforward structures to the batching problem. By means of these structures, the composition of the batches is changed. Either an order is moved from one batch to another batch (shift) or two orders contained in different batches are exchanged (swap). These structures can define movements on the same picker as well as operations including two pickers. Combination of these characteristics (swap or shift and one picker or two pickers) gives rise to four different neighborhood structures which have been used in the following sequence:

- $\mathcal{N}_3$: an order is moved from one batch to another batch assigned to the same picker;
- $\mathcal{N}_4$: an order is moved from one batch to another batch assigned to another picker;
- $\mathcal{N}_5$: two orders are exchanged which are included in different batches assigned to the same picker;
- $\mathcal{N}_6$: two orders are exchanged which are included in different batches assigned to different pickers.

We note that only neighbors are considered which represent feasible solutions, i.e. which do not violate the capacity constraint of the picking device. If an order cannot be added to a batch, this order is assigned to a new batch, where the position of the new batch is chosen such that the total tardiness is minimal.

6. Numerical experiments

6.1. Test problem instances

Chen et al. (2015) dealt with a problem which is almost identical to the one discussed here. Unfortunately, the problem instances from their experiments were not available. Therefore, we generated instances with the same characteristics and noticed that all of them could be solved to optimality within a few seconds by means of the model formulation proposed in Section 4. This can be explained by the fact (i) that the problem instances used by Chen et al. (2015) are rather small ones which only included at most 8 customer orders and (ii) that the problem itself is more simple since the authors assumed that only one order picker is available for processing the orders from which follows that no decisions concerning the assignment of batches to pickers have to be made. We, therefore, decided not to use these instances for the evaluation of the solution quality of our VND but design our own, more challenging test problem sets instead.

For our experiments, we adapted the generation of test problems from the numerical experiments of Henn (2015) who dealt with very large instances for the JOBASP. We consider a warehouse with 10 picking aisles, where the number of blocks q varies between 1 and 3, resulting in 10, 20 or 30 subaisles. Each subaisle contains 50 storage locations (25 on each side of the subaisle). The depot is located in front of the leftmost picking aisle, and the distance between the depot and the leftmost picking aisle amounts to 1.5 length units (LUs). For entering or leaving a subaisle, the order picker has to cover a distance of 1 LU. This is also the distance between two adjacent storage locations of a subaisle. The distance between two adjacent picking aisles amounts to 5 LUs.

For the assignment of articles to storage locations, a class-based storage assignment policy is applied, i.e. articles with high demand frequencies are assigned to storage locations near the depot. We use the same approach as Henn (2015), who divided the articles into three classes A, B and C, whereupon class A includes 10% of all articles with the highest demand frequency, representing up to 52% of the total demand; class B contains 30% of all articles responsible for 36% of the demand. The remaining articles, assigned to class C, are characterized by rather low demand frequencies. Within each class, articles are randomly assigned to the storage locations of the corresponding subaisles. The determination of the subaisles is based on the distance to the depot. Class A articles are assigned to storage locations in a subaisle representing 10% of all subaisles with the shortest distance to the depot. The subaisles which belong to 60% of all subaisles farthest from the depot include class C articles.

The number of customer orders is fixed to 100 and 200, whereupon the number of items contained in an order is uniformly distributed over the set $\{5, 6, \ldots, 25\}$. Each order should be completed until a certain due date. The due dates are generated based on the processing times $\tilde{u}_n$ $(n \in N)$ of the orders, the number of available order pickers $p_{\max}$ and the so-called modified traffic congestion rate (MTCR) γ describing the tightness of the due dates (Elsayed & Lee, 1996). The interval from which due dates are randomly chosen is determined as follows: $[\min\{\tilde{u}_n \mid n \in N\}, (2 \cdot (1-\gamma) \cdot \sum_{n\in N} \tilde{u}_n + \min\{\tilde{u}_n \mid n \in N\})/p_{\max}]$. As can be seen, the tightness of the due dates is dependent on the processing time $\tilde{u}_n$ of an order n, while the processing time is determined by the sequence according to which the corresponding items are to be retrieved, which is the main reason why we do not use the same instances as Henn (2015). They applied the simple S-shape and largest gap strategies for the determination of the processing times. Since we integrate the routing problem into our approach, we generate much shorter tours. This results in considerable shorter batch processing times, which is why an application of our approach to the instances of Henn (2015) results in a total tardiness of 0 for most instances. Therefore, we decided to use the LKH heuristic for the determination of the processing times $\tilde{u}_n$ $(n \in N)$ instead, which is more appropriate as the LKH heuristic is also applied within our VND algorithm. The MTCR γ has been fixed to 0.6, 0.7 and 0.8 as done by Henn (2015).

In our experiments, 2, 3 or 5 pickers are available for processing the orders. The capacity C of the picking device is set to 45 or 75 items. The time a picker needs for performing a tour is composed of the setup time, the search and pick time, and the travel time (see Section 2). The setup time amounts to 3 minutes, while searching and picking an item requires 20 seconds, and the order picker moves 20 LU per minute.

Combination of the above-mentioned parameter values results in 108 problem classes. For each problem class, 30 instances have been generated, i.e. 3240 problem instances have been solved in total. The experiments have been carried out on a desktop PC with a 3.4 gigahertz Pentium processor and 8 gigabytes RAM. The solution approach has been encoded in C++ using Microsoft Visual Studio 2015.

6.2. Evaluation of the components of the VND approach

6.2.1. Initial solution

Pretests have shown that the amount of computing time required for applying the VND is strongly dependent on the quality of the initial solution. Starting with a low-quality solution, a large number of iterations has to be carried out until a local optimum is found. Henn (2015) used the ESDR-based algorithm in order to generate an initial solution. However, this rule may result in very poor solutions since the orders are grouped into batches without considering the resulting processing times. Therefore, we proposed a seed algorithm in which both the due dates of the orders and the tour length of the corresponding batches are taken into consideration.

The computing times of both approaches are below one second if the combined heuristic is used for determining the processing times of the respective tours. Since such computing times can be neglected, both approaches are applied and the solution with the smaller total tardiness is used as the initial solution. The impact on the quality of the initial solution is depicted in Tables 2 and 3.

In Tables 2 and 3, for the problem classes with 100 and 200 orders, the quality of the initial solutions (in terms of the aver-

Table 2
Evaluation of the initial solution for 100 orders.

			2 pickers			3 pickers			5 pickers		
C	γ	q	**ESD**	**INI**		**ESD**	**INI**		**ESD**	**INI**	
			tar	tar	imp	tar	tar	imp	tar	tar	imp
45	0.6	1	130	130	0.0	112	112	0.0	89	89	0.0
45	0.6	2	70	70	0.0	165	158	3.7	128	128	0.0
45	0.6	3	235	220	6.3	347	283	18.4	517	516	0.2
45	0.7	1	1831	1017	44.4	1451	1102	24.0	1209	986	18.4
45	0.7	2	2479	1345	45.8	2245	1708	23.9	1483	1280	13.7
45	0.7	3	5347	2715	49.2	3908	2435	37.7	2808	2147	23.5
45	0.8	1	8117	4988	38.6	6181	3996	35.4	4729	3294	30.3
45	0.8	2	10,468	6522	37.7	7203	4734	34.3	6692	4502	32.7
45	0.8	3	14,434	8968	37.9	10,246	6677	34.8	3857	2694	30.2
75	0.6	1	62	62	0.0	85	85	0.0	116	116	0.0
75	0.6	2	63	63	0.0	99	99	0.0	130	130	0.0
75	0.6	3	91	91	0.0	154	154	0.0	270	270	0.0
75	0.7	1	141	141	0.0	166	166	0.0	268	268	0.0
75	0.7	2	181	181	0.0	340	340	0.0	312	312	0.0
75	0.7	3	431	431	0.0	489	489	0.0	686	686	0.0
75	0.8	1	2793	2473	11.5	2556	2399	6.1	2397	2250	6.1
75	0.8	2	3872	3407	12.0	2968	2742	7.6	3726	3134	15.9
75	0.8	3	6477	5120	21.0	5055	4289	15.1	1926	1778	7.7
Average			3179	2108	16.9	2432	1776	13.4	1741	1366	9.9

Table 3
Evaluation of the initial solution for 200 orders.

			2 pickers			3 pickers			5 pickers		
C	γ	q	**ESD**	**INI**		**ESD**	**INI**		**ESD**	**INI**	
			tar	tar	imp	tar	tar	imp	tar	tar	imp
45	0.6	1	158	158	0.0	181	180	0.7	180	180	0.0
45	0.6	2	130	130	0.0	115	115	0.0	270	270	0.0
45	0.6	3	753	604	19.8	345	345	0.0	419	419	0.0
45	0.7	1	6436	2755	57.2	5127	3551	30.7	3187	2784	12.7
45	0.7	2	8150	3461	57.5	5564	4212	24.3	3625	3307	8.8
45	0.7	3	18,697	6972	62.7	12,763	7692	39.7	9461	6865	27.4
45	0.8	1	33,890	19,487	42.5	22,982	14,169	38.3	14,632	9494	35.1
45	0.8	2	41,291	24,360	41.0	26,747	16,759	37.3	17,909	11,872	33.7
45	0.8	3	56,178	32,847	41.5	39,085	23,840	39.0	24,405	15,708	35.6
75	0.6	1	70	70	0.0	85	85	0.0	136	136	0.0
75	0.6	2	70	70	0.0	97	97	0.0	164	164	0.0
75	0.6	3	95	95	0.0	158	158	0.0	267	267	0.0
75	0.7	1	177	177	0.0	190	190	0.0	267	267	0.0
75	0.7	2	375	375	0.0	235	235	0.0	286	286	0.0
75	0.7	3	478	478	0.0	618	618	0.0	755	755	0.0
75	0.8	1	11,039	8705	21.1	7982	7454	6.6	5785	5526	4.5
75	0.8	2	14,669	11,758	19.8	9334	8852	5.2	7471	7011	6.2
75	0.8	3	22,468	16,607	26.1	16,942	13,767	18.7	11,703	9974	14.8
Average			11,951	7173	21.6	8253	5684	13.4	5607	4183	9.9

age total tardiness (tar) in minutes) is depicted which is obtained after the application of the ESDR-based algorithm (ESD) and after the additional application of the seed algorithm (INI). The improvement (imp) [in %] amounts to zero, if the additional application of the seed algorithm has no impact on the quality of the initial solutions. In these cases, solutions constructed by means of the ESDR-based algorithm always lead to a smaller total tardiness for the instances of the corresponding problem class. In fact, this is the case for instances with a very small MTCR ($\gamma = 0.6$) and with a medium MTCR ($\gamma = 0.7$) and a large capacity of the picking device ($C = 75$). These instances are characterized by quite loose due dates which can be met easily. (Note that the generation of the due dates is independent of the capacity of the picking device and due dates can be satisfied easier when the capacity is large since the total processing time of all orders decreases.) If the due dates are not tight, minimizing the processing times gets much less important since the due dates of most orders can be met by processing the orders in the sequence of non-descending due dates. This is exactly what the ESDR-based algorithm guarantees, which is the reason why the application of this rule leads to rather good solutions in these cases.

With an increasing MTCR, the due dates get tighter and harder to meet, resulting in a dramatic increase of the average total tardiness. In case of very tight due dates, it is not sufficient to find a reasonable sequence according to which the orders are to be processed. Instead, minimizing the processing times gets pivotal.

Table 4
Impact of the neighborhood structures for the case of five pickers and 200 orders.

C	γ	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$
45	0.6	25.9	37.8	6.0	11.6	0.9	17.8
45	0.7	42.9	5.8	28.3	8.7	6.3	8.1
45	0.8	33.4	2.1	7.0	9.5	35.1	12.9
75	0.6	3.9	76.0	4.1	6.4	0.2	9.3
75	0.7	12.9	36.8	8.6	18.2	0.9	22.5
75	0.8	26.2	11.1	22.4	12.9	14.8	12.6
Average		24.2	28.3	12.7	11.2	9.7	13.9

Since the processing times are not taken into consideration, solutions obtained by means of the ESDR-based algorithm are expected to be of very poor quality. The results from the numerical experiments confirm this expectation and demonstrate that the seed algorithm outperforms the ESDR-based algorithm by far when a high MTCR is assumed. By means of the seed algorithm, the average total tardiness can be decreased by up to 62.7% (200 orders, 2 pickers, $C = 45,\ \gamma = 0.7,\ q = 3$). As expected, the savings obtained by application of the seed algorithm tend to get larger with a decreasing capacity of the picking device or an increasing MTCR. For problem classes with a small capacity ($C = 45$) and a large MTCR ($\gamma = 0.8$), the reduction of the average total tardiness ranges between 30.2% (100 orders, 5 pickers, $q = 3$) and 42.5% (200 orders, 2 pickers, $q = 1$). Furthermore, it should be noted that the improvements get smaller with an increasing number of order pickers available. This can be deduced to the fact that dealing with the sequencing problem gets more important when a larger number of pickers, i.e. more possibilities of assigning orders, have to be taken into account.

As an intermediate summary, it can be stated that the application of two constructive approaches leads to significant improvements with respect to the total tardiness. On average, across all problem classes the total tardiness can be reduced by 14.2% without a noticeable increase of the computing time.

6.2.2. Neighborhood structures

When designing a VND, another very important issue refers to the choice of the neighborhood structures and the sequence according to which they are used within the algorithm. The VND approach proposed by Henn (2015) performed quite well for the JOBASP. The author suggested utilization of the batch-related neighborhood structure $\mathcal{N}_1$ and the order-related structures $\mathcal{N}_3$ to $\mathcal{N}_6$. We also use these structures and the corresponding sequence. Furthermore, we introduce a new neighborhood structure $\mathcal{N}_2$ in order to be able to break up a complete batch and reinsert the orders into other batches. The impact of the moves applied according to the neighborhood structures is depicted in Table 4.

For problem classes with 5 order pickers and 200 customer orders, Table 4 includes information about the average proportion [in %] of the improvement obtained within the local search phases according to the neighborhood structure $\mathcal{N}_l$ ($l \in \{1, \ldots, 6\}$). For example, the entry 25.9 ($C = 45,\ \gamma = 0.6,\ l = 1$) means that 25.9% of the total improvement obtained by the VND can be attributed to the local search phases within the first neighborhood structure $\mathcal{N}_1$. The average proportion of the improvement ranges from 9.7% ($\mathcal{N}_5$) to 28.3% ($\mathcal{N}_2$). Therefore, it can be concluded that all neighborhood structures should be integrated into the algorithm.

The batch-related neighborhood structure $\mathcal{N}_1$ is responsible for 24.2% of the total improvement. It can be observed that the impact of this structure is much higher than the impact of the order-related neighborhood structures $\mathcal{N}_3$–$\mathcal{N}_6$, whose proportion of the improvement amounts to approximately 10%, respectively. This can be traced back to the fact that the batch-related neighborhood structure is the first structure of the sequence and applied much more often than order-related structures. Nevertheless, the impact of structures $\mathcal{N}_3$–$\mathcal{N}_6$ should not be underestimated as improvements found in the respective local search phases may allow the algorithm to further improve the solution by continuing with $\mathcal{N}_1$ and $\mathcal{N}_2$.

The newly proposed neighborhood structure $\mathcal{N}_2$ shows the largest proportion (28.3%) of the total improvement, although it is sequenced after $\mathcal{N}_1$. The proportion strongly fluctuates, ranging from 2.1% ($C = 45,\ \gamma = 0.8$) to 76.0% ($C = 75,\ \gamma = 0.6$). The reason can be found in the generation of the initial solution. As shown in the previous subsection, the ESDR-based algorithm leads to good solutions for loose due dates which are easy to meet (instances with a low MTCR γ or a medium MTCR and a large capacity C), while it is outperformed by the seed algorithm when the due dates get tight. A move according to $\mathcal{N}_2$ is defined by breaking up a complete batch and reassigning the orders to other batches. If the initial solution is constructed by means of the seed algorithm, these moves will usually not lead to improvements since the orders are already grouped into batches in a reasonable manner which also takes into account the processing times. This is not true for solutions generated by applying the ESDR-based algorithm. In this approach, batches are constructed considering the due dates of the orders only, which is the reason why the impact of $\mathcal{N}_1$ (which simply changes the position of batches) is quite small. However, by means of moves according to $\mathcal{N}_2$, the batching can be optimized regarding the processing times resulting in massive improvements with respect to the tardiness of all orders.

6.3. Evaluation of the performance of the VND approach

6.3.1. The VND and an exact approach applied to small and medium-sized problem instances

In order to evaluate the solution quality of the proposed VND algorithm, the objective function values of the solutions provided by this approach could be compared to the ones of an exact solution approach. For this purpose, we generated small ($n \in \{10, 20\}$ orders) and medium-sized ($n = 50$ orders) instances with $p = 2$ or $p = 5$ order pickers. For each problem class, 10 instances have been generated and solved by means of the newly-proposed model formulation. Application of the IP-solver has been terminated after 7200 seconds. Problematic is the fact that solving the LP-relaxation of the model almost always results in a total tardiness equal to 0. Also, no other procedures are known for the determination of adequate lower bounds for the JOBASRP (Henn, 2015). Consequently, optimality gaps cannot be obtained. For an evaluation of the performance of the VND algorithm, we considered instances with MTCRs of $\gamma = 0.6$ and $\gamma = 0.8$ and compared the respective VND solution to the best solution found by the IP-solver (applied to the model) within 2 hours of computing time.

In Table 5, the average total tardiness per problem instance is presented for solutions generated by the IP-solver as well as for solutions provided by the VND algorithm. Furthermore, w.r.t. the VND algorithm, the average improvement (imp) [in %] of the objective function value (tardiness) in comparison to the best objective function value obtained by the IP-solver is given. Finally, average computing times are depicted for both approaches. The results demonstrate that the VND algorithm outperforms the exact approach by far, both in terms of solution quality and computing time. What concerns small instances with $n = 10$, the exact approach generated solutions of acceptable quality. In fact, for a low MTCR ($\gamma = 0.6$), three instances have been solved to optimality (two instances for $p = 2$ and one instance for $p = 5$). For such small problem instances, though, assignment and sequencing decisions are rather trivial since the number of batches is not larger than the number of pickers. Consequently, the total tardi-

Table 5
Results for the exact approach and for the VND algorithm.

n	p	γ	**Exact approach**		**VND**		
			time	tar	time	tar	imp
10	2	0.6	5822	9	2	8	8.7
10	2	0.8	7200	75	3	70	6.7
10	5	0.6	6600	26	3	23	11.4
10	5	0.8	7200	77	3	74	4.2
20	2	0.6	7200	67	8	25	63.3
20	2	0.8	7200	307	14	220	28.5
20	5	0.6	7200	63	14	34	45.6
20	5	0.8	7200	206	12	166	18.9
50	2	0.6	7200	17,249	31	34	99.7
50	2	0.8	7200	20,342	98	1167	93.6
50	5	0.6	7200	6094	33	52	99.0
50	5	0.8	7200	7161	85	622	90.6

ness is minimized by assigning at most one order to each picker. Nevertheless, even for these small instances, the exact approach was outperformed by the VND algorithm. The latter did not only manage to find optimal solutions for the three instances which have been solved optimally by the exact approach; in addition to that it provided solutions with a smaller average total tardiness for all four problem classes, where the improvements ranged between 4.2% and 11.4%. When the number of customer orders is increased, it can be observed that the solution quality of the exact algorithm drastically deteriorates. For medium-sized instances ($n = 50$), the exact approach produced solutions of inferior quality and application of the VND algorithm led to improvements ranging between 90.6% ($p = 5,\ \gamma = 0.8$) and 99.7% ($p = 2,\ \gamma = 0.6$).

We conclude that the discussed exact solution procedure, i.e. applying a commercial IP-solver to the respective model formulation, does by no means represent a competitive or even promising approach to the JOBASRP. Even for very small problem instances, the exact approach is outperformed by the proposed VND algorithm.

6.3.2. Generation of upper bounds for large problem instances

Since we will not be able to evaluate the solution quality of the proposed VND algorithm by comparing the objective function values of the provided solutions to the respective optimal objective function values or to an adequate lower bound, we will have to use upper bounds from the application of heuristic approaches as a reference instead. Three approaches will be used for generating upper bounds.

In the first approach, solutions are provided by the ESDR-based algorithm in combination with the S-shape routing strategy. The ESDR-based algorithm, which has also been used by Henn (2015), is a very straightforward approach to solve the JOBASRP. Solutions from this relatively simple approach and the corresponding objective function values will be used in order to identify what the benefits are when turning to a more complex approach (i.e. to our VND algorithm).

In practice, the respective subproblems of the JOBASRP are solved in sequence. We, therefore, designed a sequential solution approach in which state-of-the-art algorithms have been integrated for solving these subproblems. The results from this approach will be compared to those from the VND algorithm in order to identify the benefits from dealing with the JOBASRP as a holistic problem. In the first step of the sequential approach, each order is contained in a single batch; the corresponding processing times are determined on the basis of the S-shape strategy. Then, the BASP is to be solved, which results in an assignment of customer orders to order pickers. As mentioned in Section 3, when each batch includes one order only, the BASP is equivalent to the Parallel Machine Scheduling Problem with the objective of minimizing the total tardiness of all orders. We have implemented the solution approach of Biskup, Herrmann, and Gupta (2008) here which currently represents the state-of-the-art algorithm for this problem (Ullrich, 2013). In the second step, given the previously-generated assignment, for each order picker, the JOBASP is solved by the iterated local search (ILS) algorithm of Henn and Schmid (2013). The ILS algorithm is terminated after a certain time limit has been reached. The time limit t^{ILS} is chosen in such a way that the computing time of the sequential approach coincides with the time t^{VND} needed for applying the VND algorithm. Since solving a single problem instances requires for applying the ILS algorithm to the JOBASP of each order picker, the time limit of the ILS is set to $t^{\text{ILS}} = t^{\text{VND}}/p$, where p denotes the number of pickers. Finally, in the third step, in order to reduce the processing times of the batches, the LKH heuristic is applied to each of the routing problems arising from the customer orders included in the batches.

As a third upper bound, we use the initial solution to which the VND in combination with the LKH heuristic is applied. This bound is chosen in order to evaluate the improvements which are obtained by application of the VND algorithm.

6.3.3. Solution quality of the VND algorithm for large instances

In Tables 6 and 7, the average total tardiness (tar) in minutes is depicted for the ESDR-based algorithm combined with S-shape routing (ESD), the sequential solution approach (SEQ), the initial solution after the application of the LKH heuristic (INI) and the complete variable neighborhood descent algorithm (VND) for problem classes with 100 and 200 orders, respectively. Furthermore, the average improvements (imp_i) [in %] are presented compared to the total tardiness provided by approach i ($i \in \{1, 2, 3\}$) from the previous subsection.

Comparison with the ESDR-based algorithm: When comparing solutions obtained by the ESDR-based algorithm with VND solutions, significant improvements regarding the total tardiness of all customers can be observed. On average, the reduction of the total tardiness ranges from 51.5% (200 orders, 2 pickers, $C = 75,\ \gamma = 0.6,\ q = 1$) to 95.2% (200 orders, 3 pickers, $C = 45,\ \gamma = 0.7,\ q = 2$). The number of pickers does not seem to have a strong impact on the amount of improvement as the average reduction is between 69.2% and 71.2% for 100 customer orders and ranges from 73.5% to 74.8% for 200 orders. However, the impact of the other factors under investigation, namely the number of orders, the capacity C of the picking device, the MTCR γ as well as the number of blocks q does not seem to be negligible.

The more orders have to be assigned to batches, the larger the number of solutions gets since more combinations of orders exist which can be grouped into a batch. In the ESDR-based algorithm, orders are sequentially assigned to batches not taking advantage of the larger number of possibilities. In contrast to this, the VND considers much more moves according to the neighborhood structures $\mathcal{N}_3$ to $\mathcal{N}_6$. Thus, it is not surprising that the amount of improvement increases with an increasing number of customer orders. An increasing capacity of the picking device leads to a reduction with respect to the relative improvement. A larger capacity leads to tours containing more pick locations. Since the S-shape strategy is known to perform quite well if the number of pick locations is large in comparison to the number of subaisles (Roodbergen, 2001), the difference between the S-shape strategy and the LKH heuristic in terms of solution quality is quite small in these cases, resulting in less space for improvement. The reverse line of argumentation holds for the impact of an increase in the number of blocks as more blocks result in more subaisles, leading to a smaller number of pick locations per subaisle. Thus,

Table 6
Evaluation of the VND for 100 orders.

C	γ	q	2 pickers							3 pickers							5 pickers						
			ESD	**SEQ**	**INI**	**VND**				**ESD**	**SEQ**	**INI**	**VND**				**ESD**	**SEQ**	**INI**	**VND**			
			tar	tar	tar	tar	imp_1	imp_2	imp_3	tar	tar	tar	tar	imp_1	imp_2	imp_3	tar	tar	tar	tar	imp_1	imp_2	imp_3
45	0.6	1	181	141	98	54	70.1	61.6	45.1	153	137	86	46	69.8	66.2	46.4	121	136	73	43	64.6	68.6	41.9
45	0.6	2	180	186	55	36	80.0	80.7	34.7	340	230	122	55	83.7	75.9	54.6	281	216	100	54	80.9	75.1	46.3
45	0.6	3	525	204	107	65	87.5	67.9	39.1	716	318	156	87	87.8	72.6	44.2	1052	415	236	148	85.9	64.3	37.2
45	0.7	1	2754	909	840	359	87.0	60.5	57.3	2061	743	906	312	84.9	58.0	65.6	1621	712	824	323	80.0	54.5	60.8
45	0.7	2	4647	1494	1140	545	88.3	63.5	52.2	3712	1303	1452	665	82.1	49.0	54.2	2415	1012	1099	451	81.3	55.4	58.9
45	0.7	3	7751	2336	1606	995	87.2	57.4	38.1	5544	1598	1605	741	86.6	53.6	53.8	3886	1414	1535	749	80.7	47.0	51.2
45	0.8	1	9145	4586	4614	3590	60.7	21.7	22.2	6880	3439	3723	2826	58.9	17.8	24.1	4301	2322	2526	1873	56.4	19.3	25.8
45	0.8	2	12,773	6293	6076	4859	62.0	22.8	20.0	8730	4136	4422	3345	61.7	19.1	24.3	5705	2978	3107	2362	58.6	20.7	24.0
45	0.8	3	16,987	7628	7123	5750	66.1	24.6	19.3	11,980	5473	5363	4306	64.1	21.3	19.7	7780	3674	3716	2892	62.8	21.3	22.2
75	0.6	1	71	63	57	33	53.2	47.2	42.1	96	62	77	35	63.0	43.0	54.2	130	81	107	35	73.3	56.7	67.5
75	0.6	2	81	61	58	30	62.5	50.1	47.8	122	98	90	42	65.4	57.1	53.3	167	109	118	46	72.5	57.8	60.9
75	0.6	3	119	78	65	41	65.6	47.8	37.4	203	118	100	53	74.0	55.2	47.4	350	219	193	109	68.9	50.3	43.6
75	0.7	1	174	140	127	83	52.4	40.9	34.5	211	195	142	84	60.3	56.9	40.8	319	294	235	143	55.3	51.4	39.4
75	0.7	2	289	220	153	105	63.8	52.4	31.3	528	362	279	171	67.7	52.7	38.6	462	381	270	161	65.1	57.6	40.2
75	0.7	3	750	365	205	154	79.5	57.9	24.9	798	423	268	187	76.5	55.8	30.1	1083	551	377	257	76.3	53.4	32.0
75	0.8	1	3397	1537	2204	1170	65.6	23.9	46.9	2969	1567	2206	1211	59.2	22.7	45.1	2193	1321	1672	931	57.6	29.6	44.3
75	0.8	2	5319	2366	3066	1737	67.3	26.6	43.3	3961	1835	2446	1315	66.8	28.3	46.2	3046	1727	2077	1200	60.6	30.5	42.3
75	0.8	3	8238	3160	3792	2294	72.2	27.4	39.5	6243	2602	3333	1972	68.4	24.2	40.8	4487	2149	2558	1563	65.2	27.2	38.9
Average			4077	1765	1744	1217	70.6	46.4	37.5	3069	1369	1488	970	71.2	46.1	43.5	2189	1095	1157	741	69.2	46.7	43.2

Table 7
Evaluation of the VND for 200 orders.

C	γ	q	2 pickers							3 pickers							5 pickers						
			ESD	**SEQ**	**INI**	**VND**				**ESD**	**SEQ**	**INI**	**VND**				**ESD**	**SEQ**	**INI**	**VND**			
			tar	tar	tar	tar	imp_1	imp_2	imp_3	tar	tar	tar	tar	imp_1	imp_2	imp_3	tar	tar	tar	tar	imp_1	imp_2	imp_3
45	0.6	1	228	151	114	63	72.6	58.6	45.2	266	241	141	63	76.3	73.9	55.4	253	245	143	76	69.8	68.9	46.5
45	0.6	2	295	177	99	53	82.2	70.2	46.7	284	363	91	58	79.5	84.0	36.4	591	447	203	98	83.5	78.1	51.8
45	0.6	3	1978	497	232	97	95.1	80.5	58.1	1008	513	135	94	90.7	81.7	30.5	1026	629	199	130	87.3	79.3	34.6
45	0.7	1	10,219	2166	2304	671	93.4	69.0	70.9	7816	1892	2959	552	92.9	70.8	81.3	4745	1418	2224	433	90.9	69.5	80.5
45	0.7	2	16,722	3715	3038	1040	93.8	72.0	65.8	11,392	2504	3475	551	95.2	78.0	84.2	7071	1905	2491	505	92.9	73.5	79.7
45	0.7	3	28,590	5761	3980	1815	93.7	68.5	54.4	19,366	4323	5192	1455	92.5	66.3	72.0	13,585	3686	5052	1475	89.1	60.0	70.8
45	0.8	1	37,946	17,689	18,075	14,069	62.9	20.5	22.2	25,665	11,994	13,208	9535	62.8	20.5	27.8	16,298	7765	8966	6311	61.3	18.7	29.6
45	0.8	2	50,187	22,165	22,667	17,632	64.9	20.5	22.2	32,649	14,481	15,722	11,283	65.4	22.1	28.2	21,624	9855	11,260	7912	63.4	19.7	29.7
45	0.8	3	66,036	27,281	26,109	20,557	68.9	24.6	21.3	45,702	19,235	19,245	14,804	67.6	23.0	23.1	28,415	13,570	13,058	9511	66.5	29.9	27.2
75	0.6	1	78	61	63	37	51.9	38.8	40.5	98	71	78	37	62.7	48.2	53.0	156	125	123	54	65.1	56.5	55.8
75	0.6	2	90	60	63	33	62.9	45.0	47.1	123	95	89	43	65.2	54.7	51.6	213	175	148	71	66.8	59.6	52.1
75	0.6	3	139	91	63	36	73.7	59.9	42.0	204	138	114	68	66.7	50.7	40.7	340	233	196	103	69.7	55.8	47.6
75	0.7	1	229	235	146	99	56.6	57.8	31.9	242	249	163	93	61.7	62.9	43.3	316	350	236	137	56.7	60.9	42.1
75	0.7	2	560	478	326	218	61.1	54.5	33.3	361	286	193	126	65.0	55.8	34.6	437	379	246	135	69.0	64.3	44.9
75	0.7	3	924	406	221	160	82.7	60.7	27.7	1143	615	317	221	80.6	64.0	30.3	1414	810	393	264	81.3	67.4	32.7
75	0.8	1	13,381	5090	8172	3687	72.4	27.6	54.9	9484	4030	6704	2804	70.4	30.4	58.2	6763	3331	5092	2280	66.3	31.5	55.2
75	0.8	2	20,149	7281	10,792	5165	74.4	29.1	52.1	13,026	4963	7953	3148	75.8	36.6	60.4	9832	4351	6375	2913	70.4	33.1	54.3
75	0.8	3	29,080	9331	12,385	6127	78.9	34.3	50.5	21,485	7564	11,051	5156	76.0	31.8	53.3	14,462	5704	8279	3922	72.9	31.2	52.6
Average			15,379	5702	6047	3975	74.6	49.6	43.7	10573	4087	4824	2783	74.8	53.1	48.0	7086	3054	3593	2018	73.5	53.2	49.3

the solution quality of the S-shape policy deteriorates and the amount of the improvement obtained by applying the VND (into which two more sophisticated routing algorithms are integrated) increases. The MTCR determines how tight the due dates are. A larger MTCR leads to tighter due dates which tend to increase the total tardiness of solutions to the corresponding problem instance. Apart from problem classes with an MTCR $\gamma = 0.8$ and a capacity $C = 45$, the amount of improvement increases with an increasing MTCR. This can be explained by the fact that the solution quality of the ESDR-based algorithm deteriorates when the due dates get tighter (see Section 6.2.1). If the MTCR is low or medium ($\gamma \in \{0.6, 0.7\}$), the VND leads to solutions with an average total tardiness of up to 1815 minutes (200 orders, 2 pickers, $C = 45$, $\gamma = 0.7$, $q = 3$), i.e. the average tardiness of an order amounts to 9 minutes. For instances of these problem classes, the VND provides many solutions with a total tardiness which is zero or close to zero, resulting in improvements of 100% (regardless of the absolute amount of improvement). Thus, conclusions have to be drawn carefully for instances with a small MTCR. If the MTCR is large ($\gamma = 0.8$), the average total tardiness significantly increases. Finding solutions with a low total tardiness gets even harder when the capacity is small ($C = 45$) since, the total processing times of the orders increase. Therefore, the amount of the relative improvement decreases when very difficult instances are considered. Nevertheless, the VND manages to massively reduce the total tardiness as an absolute improvement of up to 45,479 minutes (200 orders, 2 pickers, $C = 45$, $\gamma = 0.8$, $q = 3$) is achieved, which corresponds to a decrease of the tardiness by approximately 4 hours per order.

Comparison with the sequential solution approach: In the VND algorithm, neighborhood structures deal with the assignment and sequencing problem, and with the batching problem, while routing algorithms are integrated for the determination of processing times and for the evaluation of solutions. Thus, all three subproblems are considered jointly in the VND approach. The resulting holistic problem is very complex and difficult to solve. Therefore, we evaluate the benefit from using the integrated approach compared to solving the subproblems in sequence by applying a state-of-the-art algorithm to each subproblem, which is done in the second approach for generating upper bounds.

The results of the numerical experiments indicate that solving the subproblems jointly reduces the total tardiness by 49.2% on average across all problem classes. This clearly demonstrates that solving these subproblems simultaneously is pivotal for achieving high-quality solutions. The number of order pickers available for processing customer orders as well as the number of blocks in the picking area do not seem to have a large impact on the amount of improvement. The impact of the number of customer orders is also quite small. For 100 customer orders, the joint solution approach is able to decrease the total tardiness by 46.4%, while the relative improvement amounts to 52.0% on average for instances containing 200 orders. The largest impact on the solution quality of the sequential approach can be observed for the MTCR γ and the capacity C. If the MTRC is quite small ($\gamma = 0.6$), using the joint approach results in a reduction of the total tardiness by up to 84.0% (200 orders, 3 pickers, $C = 45$, $q = 2$). In this case, due dates of most customer orders can be met quite easily. This is verified by the VND algorithm which manages to find solutions, where the average tardiness of a customer order is less than a minute. Such high-quality solutions are rarely obtained when applying the sequential approach. This stems from the large objective function values of solutions generated at an early stage of the solution process. In the first step of the sequential approach, orders are assigned to order pickers and the simple S-shape strategy is used to determine processing times. The batching and the routing problems are not considered by the assignment algorithm at all. It only ensures that the sum of the processing times of the orders is fairly evenly distributed across all pickers. Consequently, nearly all customer orders get delayed significantly, where the average tardiness of an order amounts to more than 140 minutes. Based on the generated assignment, the algorithms applied for subsequent batching and routing are not capable of providing high-quality solutions to the joint problem. With an increasing MTCR, assignment decisions get less important as it is not possible to meet the due dates of the orders. When orders are tardy anyway, it is more pivotal to find good solutions to the batching and routing problems in order to decrease the processing times. Nevertheless, even for a large MTCR ($\gamma = 0.8$), the VND clearly outperforms the sequential approach, reducing the total tardiness by at least 17.8% (100 orders, 3 pickers, $C = 45$, $q = 1$).

Depending on the MTCR, an increasing capacity has different impacts on the amount of improvement. If the MTCR is small ($\gamma = 0.6$), increasing the capacity of the picking device implies less improvement by the VND. As pointed out before, in case of a small MTCR, the sequential approach leads to solutions of very poor quality because of bad assignment decisions. However, when the capacity of the picking device is large, the subsequent batching algorithm has lots of options to improve the solution, making the assignment decisions to have less impact on the solution quality. For a large MTCR ($\gamma = 0.8$), the relative improvement obtained by the VND increases with an increasing capacity. Whereas the impact is almost negligible for a small number of customer orders and order pickers, the difference gets very large when problems including 200 orders are considered. As mentioned before, finding high-quality solutions to the batching and routing problems for reducing processing times is pivotal in those instances, while the solution space gets quite large with an increasing number of orders and capacity of the picking device. Starting from a very poor solution, where each order is contained in a single batch, the ILS algorithm dealing with the batching subproblem is not able to reach the part of the search space containing very promising solutions within a small amount of computing time. (Note that the time limit for the ILS is smaller than the computing time required for applying the VND as the ILS has to be applied to the JOBASP of each picker.)

Comparison with the initial solution of the VND algorithm: In order to investigate the performance of the VND, we also compare the solutions to the initial solution in combination with the LKH heuristic. It can be observed that the VND manages to reduce the total tardiness by between 19.3% (100 orders, 2 pickers, $C = 45$, $\gamma = 0.8$, $q = 3$) and 84.2% (200 orders, 3 pickers, $C = 45$, $\gamma = 0.7$, $q = 2$). On average, across all problem classes, the improvement amounts to 44.2%, which demonstrates that the application of the VND is pivotal for the generation of high-quality solutions. Three main factors can be identified which have an impact on the amount of the improvement obtained. First, as observed in the comparison to the ESDR-based algorithm, a larger number of orders allows for more space for improvement. Second, the amount of improvement tends to decrease with an increasing number of blocks. Third, the combination of the MTCR γ and the capacity C has a great impact. While the largest improvements can be obtained for $\gamma = 0.7$ and $C = 45$, the smallest reductions are observed for $\gamma = 0.8$ and $C = 45$. This can be explained by the way how the initial solution is generated. The combination $\gamma = 0.8$ and $C = 45$ results in instances with very tight due dates which are very difficult to meet. In this case, the seed algorithm leads to very good solutions and, therefore, is chosen for the construction of initial solutions. Due to the high quality of these solutions, only small improvements can be obtained by application of the VND. In contrast, the combination $\gamma = 0.7$ and $C = 45$ leads to due dates not tight enough for the seed algorithm but too tight for the ESDR-based algorithm. Thus, the quality of the initial solution is quite

Table 8
Computing time (in seconds) required by the VND algorithm.

C	γ	q	100 orders			200 orders		
			Number of pickers			Number of pickers		
			2	3	5	2	3	5
45	0.6	1	77	80	96	222	349	448
45	0.6	2	52	70	68	164	229	501
45	0.6	3	48	72	88	256	252	474
45	0.7	1	293	377	416	1494	2553	3091
45	0.7	2	198	264	289	1130	2022	2680
45	0.7	3	160	252	268	1348	2597	3347
45	0.8	1	342	371	367	2263	2990	3621
45	0.8	2	216	260	259	1951	2264	2875
45	0.8	3	204	247	292	1918	2488	3155
75	0.6	1	122	126	151	231	255	334
75	0.6	2	46	49	57	101	121	184
75	0.6	3	50	59	72	106	141	218
75	0.7	1	184	164	243	406	449	518
75	0.7	2	72	104	100	185	209	260
75	0.7	3	85	103	120	209	231	388
75	0.8	1	545	685	596	2472	3075	3044
75	0.8	2	276	319	310	1629	1820	1940
75	0.8	3	338	431	376	1775	2027	2284
Average			184	224	232	992	1337	1631

poor and the VND manages to significantly reduce the total tardiness.

6.3.4. Considerations regarding computing times

Apart from the solution quality, the VND is evaluated with respect to the computing times required. Henn (2015) proposed solution approaches to the JOBASP using very simple routing strategies and reported computing times of up to 25 minutes for problem instances with 200 orders. It can be expected that our solution approach requires a higher computational effort as we integrated more complex routing algorithms. Furthermore, Henn (2015) considered a single-block layout only. Especially when the capacity is large and the simple S-shape strategy is used, many batches will result in the same tours as all picking aisles will be traversed. Due to this characteristic, the problem is reduced to a sequencing problem and solution approaches will terminate much faster as there is less room for improvement. This is not true for our setup since we also consider more complex layouts. Furthermore, tours constructed by the LKH heuristic are dependent on the certain pick locations instead of the subaisles to be visited only as it is the case for the S-shape strategy.

In Table 8, the average computing time required by the VND for solving a problem instance is depicted for all problem classes. As can be seen, computing times are dependent on the number of customer orders as well as on how difficult the due dates can be met. Of course, a larger number of orders significantly increases the computing times as it allows for much more possible moves regarding the neighborhood structures. For problem classes with 100 orders, the computing times are not a critical issue as they range from 1 to 11 minutes. When 200 orders are considered, however, up to 1 hour is required for solving a single instance. The largest computing times can be observed for problems with a large MTCR ($\gamma = 0.8$) and a small capacity ($C = 45$), which can be explained by the fact that these instances are characterized by very tight due dates. In practical applications, such high computing times may be a critical issue, which is why we investigate to which extent the solution quality decreases if less computing time is spent. Since the largest computing times arise for instances with

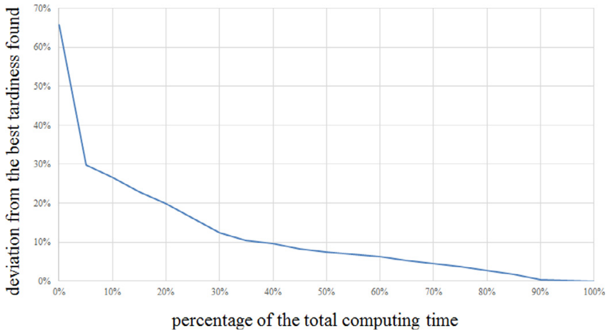


Fig. 7. Development of the solution quality over time.

200 orders, 5 pickers, $C = 45$, $\gamma = 0.8$ and $q = 1$, we depicted the development of the solution quality over time for these instances in Fig. 7.

The VND starts with an initial solution in which the combined heuristic is used in order to determine the processing times of the batches. The computing time for constructing this solution is negligible. Then, the LKH heuristic is applied, significantly reducing the processing times and improving the objective function value. On average, application of the LKH heuristic to all batches only requires 12 seconds of computing time for instances from this problem class. The remaining improvement is obtained by the moves corresponding to the different neighborhood structures. As can be seen in Fig. 7, the largest proportion of the improvement is realized within the first 30% of the computing time. After investing 40% of the time, which corresponds to 24 minutes, the total tardiness is 10% above the tardiness of the best solution found after the VND terminates. However, it can be observed that the VND manages to steadily improve the solution until the end of the solution process, i.e. reducing the computing time will definitely result in a larger total tardiness.

If the solution process has to be speeded up, the removal of $\mathcal{N}_6$ would be a straightforward way as this neighborhood structure requires the highest computational effort. Considering instances from the problem class defined above, a removal of $\mathcal{N}_6$ would save 44.0% of the total computing time, while the average total tardiness would increase by 6.4%. Another possible modification for saving computing time consists in the application of other routing algorithms. In order to determine the processing times of the batches, which is necessary to evaluate a solution, a routing problem has to be solved for each batch. This is done by means of the combined heuristic which represents a very fast solution approach. However, whenever a local optimum has been found, the processing times of the batches are improved by applying the LKH heuristic. This step consumes a large proportion of the total computing time required by the VND. Therefore, the impact of integrating routing algorithms, which are based on more simple improvement strategies (e.g. 2-opt or 3-opt), is evaluated in Table 9.

Table 9 depicts the relative average deviation of the total tardiness (Δtar) [in %] as well as the deviation of the computing time (Δtime) [in %] when applying the 2-opt or 3-opt improvement strategies to a local optimum instead of using the LKH heuristic. Only problem classes with 200 orders, 5 pickers, an MTCR γ equal to 0.8 or with $\gamma = 0.8$ and a capacity C of 45 are considered here, as instances from these classes clearly require the highest amount of computing time when being solved by means of the VND algorithm. As can be seen from the table, the tardiness increases while computing times decrease when using more simple improvement strategies. On average, over all problem classes under consideration, the tardiness increases by 15.6% (2-opt) and 5.0% (3-opt),

Table 9
Results for the application of the 2-opt and 3-opt improvement strategies instead of the LKH heuristic.

C	γ	q	LKH		2-opt				3-opt			
			tar	time	tar	Δtar	time	Δtime	tar	Δtar	time	Δtime
45	0,7	1	433	3091	555	28.1	2153	−30.3	460	6.1	2247	−27.3
45	0,7	2	505	2680	676	33.8	2068	−22.8	548	8.5	2084	−22.2
45	0,7	3	1475	3347	1735	17.6	2740	−18.1	1553	5.3	2951	−11.8
45	0,8	1	6311	3621	6821	8.1	2270	−37.3	6497	2,9	2456	−32.2
45	0,8	2	7912	2875	8519	7.7	2277	−20.8	8135	2,8	2078	−17.3
45	0,8	3	9511	3155	10,141	6,6	2536	−19.6	9728	2.3	2624	−16.8
75	0,8	1	2280	3044	2525	10.7	1618	−46.8	2373	4.1	2140	−29.7
75	0,8	2	2913	1840	3368	15.6	1225	−33.4	3114	6.9	1541	−16.2
75	0,8	3	3922	2284	4382	11.7	1783	−21.9	4161	6.1	1934	−15.3
Average			3918	2882	4302	15.6	2075	−27.9	4063	5.0	2228	−21.0

whereas the average computing time is reduced by 27.9% (2-opt) and 21.0% (3-opt). Thus, the results indicate that using more simple improvement strategies is a valid option if computing times are a critical issue. This holds for the 3-opt improvement strategy in particular. Whereas the integration of the 2-opt strategy leads to an increase of the total tardiness by up to 33.8% ($C = 45$, $\gamma = 0.7$, $q = 2$), using 3-opt results in an acceptable solution quality in all problem classes. Considering the problem class where solving instances by means of the original VND algorithm consumes the largest amount of computing time ($C = 45$, $\gamma = 0.8$, $q = 1$), applying 3-opt instead of the LKH heuristic reduces the average computing time by 32.2%, while increasing the average total tardiness by 2.9% only. Thus, we conclude that the VND can be adjusted, if necessary, in such a way that solutions with a reasonable quality can be found within an acceptable amount of computing time.

7. Conclusion

In this paper, we considered the Joint Order Batching, Assignment and Sequencing, and Routing Problem which is rarely addressed in the literature, although it is pivotal for an efficient organization of manual order picking systems. We proposed a new mathematical model formulation. In contrast to existing formulations, the size of the model increases polynomially with the number of customer orders, which allows for solving small instances within a reasonable amount of computing time. Furthermore, we designed a variable neighborhood descent approach which is able to deal with very large problems. In order to reach the local optimum quite fast and speed up the solution process, a new heuristic for the construction of an initial solution is developed which outperforms the earliest start date rule-based algorithm by far when due dates are very tight.

By means of numerical experiments, the solution quality of the variable neighborhood descent algorithm is evaluated. First, the initial solution is compared to solutions obtained by applying the earliest start date rule-based algorithm, which represents a common approach to generate a solution and was also used by Henn (2015). It is demonstrated that our initial solution leads to a reduction of the total tardiness by up to 63%. In a second step, we show that all neighborhood structures are important in order to obtain high-quality solutions. The largest proportion of the total improvement is achieved by moves regarding a newly designed neighborhood structure which breaks up complete batches and reassigns the orders to other batches. Finally, several approaches for generating upper bounds are used in order to evaluate the quality of solutions provided by the variable neighborhood descent algorithm. Combining the earliest start date rule-based algorithm with the simple S-shape strategy represents a very straightforward way to solve the problem. However, it is pointed out that the solution quality of this approach is very poor as the variable neighborhood descent manages to improve the objective function value of these solutions by up to 95%, which means a massive reduction of the total tardiness. Another upper bound is obtained by decomposing the Joint Order Batching, Assignment and Sequencing, and Routing Problem into its subproblems and sequentially solving them by means of state-of-the-art algorithms. Thus, this upper bound represents the total tardiness which results from considering the subproblems separately as done in the literature so far. Compared to this sequential solution approach, applying the variable neighborhood descent algorithm reduces the total tardiness of all customer orders by up to 84%, which demonstrates that dealing with the Joint Order Batching, Assignment and Sequencing, and Routing Problem as a holistic problem is inevitable for an efficient organization of warehouse operations.

We dealt with the Joint Order Batching, Assignment and Sequencing, and Routing Problem in warehouses with wide aisles which allow order pickers to overtake each other. Further research could concentrate on picker blocking aspects arising in narrow-aisle warehouses, where subaisles may be blocked when two pickers are moving in opposite directions or traffic jams may occur when several pickers have to visit the same storage location at the same time. When taking blocking aspects into account, tours of different pickers cannot be independently determined anymore, making the problem much more difficult to deal with. From a practical point of view, the on-line variant of the Joint Order Batching, Assignment and Sequencing, and Routing Problem would also be a very interesting topic as customer orders arrive during the day and are not known in advance, which requires for very fast solution approaches in order to be able to recalculate each time a certain number of new orders has arrived.

References

Biskup, D., Herrmann, J., & Gupta, J. N. D. (2008). Scheduling identical parallel machines to minimize total tardiness. *International Journal of Production Economics, 115*, 134–142.
Bozer, Y. A., & Kile, J. W. (2008). Order batching in walk-and-pick order picking systems. *International Journal of Production Research, 46*, 1887–1909.
Chen, T.-L., Cheng, C.-Y., Chen, Y.-Y., & Chan, L.-K. (2015). An efficient hybrid algorithm for integrated order batching, sequencing and routing problem. *International Journal of Production Economics, 159*, 158–167.
Elsayed, E., & Lee, M. K. (1996). Order processing in automated storage/retrieval systems with due dates. *IIE Transactions, 28*, 567–577.
Elsayed, E., Lee, M. K., Kim, S., & Scherer, E. (1993). Sequencing and batching procedures for minimizing earliness and tardiness penalty for order retrievals. *International Journal of Production Research, 31*, 727–738.
Elsayed, E. A. (1981). Algorithms for optimal material handling in automatic warehousing systems. *International Journal of Production Research, 19*, 525–535.
Gademann, N., van den Berg, J., & van der Hoff, H. (2001). An order batching algorithm for wave picking in a parallel-aisle warehouse. *IIE Transactions, 33*, 385–398.

Grosse, E. H., Glock, C. H., & Ballester-Ripoll, R. (2014). A simulated annealing approach for the joint order batching and order picker routing problem with weight restrictions. *International Journal of Operations and Quantitative Management, 20*, 65–83.

Hansen, P., & Mladenović, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research, 130*, 449–467.

Helsgaun, K. (2000). An effective implementation of the Lin–Kernighan traveling salesman heuristic. *European Journal of Operational Research, 126*, 106–130.

Henn, S. (2015). Order batching and sequencing for the minimization of the total tardiness in picker-to-part warehouses. *Flexible Services and Manufacturing, 27*, 86–114.

Henn, S., Koch, S., & Wäscher, G. (2012). Order batching in order picking warehouses: A survey of solution approaches. In R. Manzini (Ed.), *Warehousing in the global supply chain: Advanced models, tools and applications for storage systems* (pp. 105–137). London: Springer.

Henn, S., & Schmid, V. (2013). Metaheuristics for order batching and sequencing in manual order picking systems. *Computers & Industrial Engineering, 66*, 338–351.

de Koster, R., Le-Duc, T., & Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review. *Science Direct, 182*, 481–501.

Koulamas, C. (1994). The total tardiness problem: Review and extensions. *Operations Research, 42*, 1025–1041.

Kulak, O., Sahin, Y., & Taner, M. E. (2012). Joint order batching and picker routing in single and multiple-cross-aisle warehouses using cluster-based tabu search algorithms. *Flexible Services and Manufacturing Journal, 24*, 52–80.

Letchford, A. N., Nasiri, S. D., & Theis, D. O. (2013). Compact formulations of the Steiner traveling salesman problem and related problems. *European Journal of Operational Research, 228*, 83–92.

Petersen, C. G., & Schmenner, R. W. (1999). An evaluation of routing and volume-based storage policies in an order picking operation. *Decision Science, 30*, 481–501.

Pinedo, M. L. (2016). *Scheduling: Theory, algorithms, and systems* (5th). Springer. Cham et al.

Ratliff, H. D., & Rosenthal, A. R. (1983). Order-picking in a rectangular warehouse: A solvable case of the traveling salesman problem. *Operations Research, 31*, 507–521.

Roodbergen, K. J. (2001). *Layout and routing methods for warehouses*. Trial: Rotterdam.

Roodbergen, K. J., & de Koster, R. (2001a). Routing methods for warehouses with multiple cross aisles. *International Journal of Production Research, 39*, 1865–1883.

Roodbergen, K. J., & de Koster, R. (2001b). Routing order pickers in a warehouse with a middle aisle. *European Journal of Operational Research, 133*, 32–43.

Scholz, A., Henn, S., Stuhlmann, M., & Wäscher, G. (2016). A new mathematical programming formulation for the single-picker routing problem. *European Journal of Operational Research, 253*, 68–84.

Scholz, A., & Wäscher, G. (2017). Order batching and picker routing in manual order picking systems: The benefits of integrated routing. *Central European Journal of Operations Research, 25*, 491–520.

Theys, C., Bräysy, O., Dullaert, W., & Raa, B. (2010). Using a TSP heuristic for routing order pickers in warehouses. *European Journal of Operational Research, 200*, 755–763.

Tompkins, J. A., White, J. A., Bozer, Y. A., & Tanchoco, J. M. A. (2010). *Facilities planning* (4th). New Jersey: John Wiley & Sons.

Tsai, C.-Y., Liou, J. J. H., & Huang, T.-M. (2008). Using a multiple-GA method to solve the batch picking problem: Considering travel distance and order due time. *International Journal of Production Research, 46*, 6533–6555.

Ullrich, C. (2013). Integrated machine scheduling and vehicle routing with time windows. *European Journal of Operational Research, 227*, 152–165.

Wäscher, G. (2004). Order picking: A survey of planning problems and methods. In H. Dyckhoff, R. Lackes, & J. Reese (Eds.), *Supply chain management and reverse logistics* (pp. 323–347). Berlin: Springer.

Part VIII:

Order Picking and Vehicle Routing for the Minimization of the Total Tardiness

WORKING PAPER SERIES

Integrated Order Picking and Vehicle Routing with Due Dates

Daniel Schubert/André Scholz/Gerhard Wäscher

Working Paper No. 7/2017

OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG

FACULTY OF ECONOMICS
AND MANAGEMENT

Impressum (§ 5 TMG)
Herausgeber:
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Der Dekan

Verantwortlich für diese Ausgabe:
Daniel Schubert, André Scholz, Gerhard Wäscher
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Postfach 4120
39016 Magdeburg
Germany

http://www.fww.ovgu.de/femm

Bezug über den Herausgeber
ISSN 1615-4274

Integrated Order Picking and Vehicle Routing with Due Dates

D. Schubert, A. Scholz, G. Wäscher

Abstract

Supermarkets typically order their goods from a centrally located distribution center (warehouse). Each order that the warehouse receives is characterized by the requested items, the location of the respective supermarket and a due date by which the items have to be delivered. For processing an order, a human operator (order picker) retrieves the requested items from their storage locations in the warehouse first. The items are then available for shipment and loaded on the vehicle which performs the tour including the respective location of the supermarket. Whether and to which extent a due date is violated (tardiness) depends on the composition of the tours, the corresponding routes and the start dates of the tours (vehicle routing subproblem). The start date of a tour, however, is also affected by the assignment of orders to pickers and the sequence according to which the orders are processed by the pickers (order picking subproblem). Although both subproblems are closely interconnected, they have not been considered simultaneously in the literature so far. In this paper, an iterated local search algorithm is designed for the simultaneous solution of the subproblems. By means of extensive numerical experiments, it is shown that the proposed approach is able to generate high-quality solutions even for large instances. Furthermore, the economic benefits of an integrated solution are investigated. Problem classes are identified, where the sequential solution of the subproblems leads to acceptable results, and it is pointed out in which cases an integrated solution is inevitable.

Keywords: Vehicle Routing, Order Picking, Parallel Machine Scheduling, Iterated Local Search

Corresponding author:
André Scholz
Otto-von-Guericke University Magdeburg, Faculty of Economics and Management
Postbox 4120, 39016 Magdeburg, Germany
Phone: +49 391 67 51841
Fax: +49 391 67 48223
Email: andre.scholz@st.ovgu.de

1 Introduction

Supermarkets are typically supplied once per workday (DVZ, 2013) from a centrally located distribution center (warehouse) with perishable goods. When the warehouse has received a certain number of orders from the supermarkets, the orders are assigned to human operators (order pickers) who retrieve the respective items from their storage locations. Each order picker processes the orders one by one in a particular sequence. All items belonging to an order are grouped together on transport devices like pallets or boxes and then, together with the items from other orders, loaded on vehicles (trucks) which deliver the goods to the respective supermarkets. This gives rise to a vehicle routing problem, namely how the items of the various orders are to be assigned to vehicles and in which sequence the supermarkets are to be visited on each tour. The solution to the vehicle routing problem determines the actual delivery dates, i.e. the points in time when each supermarket is being served. However, a vehicle can only leave from the warehouse and the respective tour can only be started when the items of all orders allocated to the tour have been retrieved completely. Thus, the actual delivery dates are also affected by the assignment of orders to order pickers and the sequence in which the orders are processed.

In practice, distribution centers and supermarkets have agreed on deadlines by which the ordered items have to be delivered. For supermarkets, complying with such deadlines is of uttermost importance as only very limited safety stocks exist and empty shelves will result in clients satisfying their demands at competitor outlets. For the distribution centers, a violation of the deadlines will, therefore, result in – often heavy – fines or even in the loss of customers if the deadlines are violated more permanently. However, due to short response times which have also been agreed between distribution centers and supermarkets, deadlines are often difficult to meet. Thus, deadlines will not only have to be considered for the determination of vehicle tours but also for the assignment of orders to order pickers and the scheduling of the orders. Consequently, dispatchers of the picking and shipping processes in warehouses are confronted with a complex decision problem. Given a set of orders from supermarkets and corresponding deadlines, it has to be decided (1) how these orders have to be assigned to order pickers, (2) how the orders assigned to each order picker have to be sequenced, (3) how the orders have to be allocated to vehicles, and (4) in which sequence the supermarkets should be visited by each vehicle such that the violation of the deadlines is minimized. We will refer to this problem as the order assignment and sequencing, and vehicle routing problem (OASVRP).

So far, both in literature and in practice of warehouse management and control, the OASVRP has not been dealt with holistically. Instead, the order assignment and the order sequencing problem on the one hand and the vehicle routing problem, on the other hand, are treated and solved separately (Schmid et al., 2013). With respect to the previously sketched interdependencies between these problems, it can be expected, though, that an integrative solution approach to the OASVRP can provide a significant source of the reduction of costs and improved customer service by allowing for an improved compliance with given delivery deadlines. Our goal, therefore, is twofold: First, we intend to present a solution approach to the OASVRP which provides high-quality solutions within an acceptable amount of computing time. Since the above-mentioned subproblems are NP-hard already, we concentrate on a heuristic solution approach. In particular, an iterated local search algorithm is proposed. This type of metaheuristic is chosen since it has already been proven to provide excellent results for other challenging optimization problems in warehouse management. Second, by means of this algorithm, we will analyze whether, under which conditions, and to what extent benefits arise from dealing with the OASVRP holistically.

Special attention will be given to the fact that in practice large problem instances have to be solved. For instance, the EDEKA group Minden-Hannover, a large cooperative of independent supermarkets in Germany, serves 1513 supermarkets from nine warehouses (EDEKA, 2017), i.e. each warehouse has to provide goods for more than 150 customers on average.

The remainder of the paper is organized as follows: In Section 2, the OASVRP will be stated in greater detail. The related literature will be reviewed in Section 3. Section 4 comprises the presentation of the proposed iterated local search approach, where the generation of an initial solution, the structure of the neighborhoods used within the improvement phase and the design of the perturbation phase will be described in particular. Section 5 is devoted to the numerical experiments which have been carried out in order to evaluate the performance of the proposed algorithm but also in order to identify the benefits resulting from a holistic approach to the OASVRP. The paper concludes with a summary and an outlook on further research (Section 6).

2 Problem description

The order assignment and sequencing, and vehicle routing problem (OASVRP), which will be described in this section, deals with picking requested items from a warehouse and delivering them to the respective customers. Let a set of orders be given, each of which specifying certain items and the corresponding demands from a particular customer. Furthermore, each order has been assigned a deadline (due date) according to which the items have to be received by the customer. The items of each order have to be shipped as a unit, split deliveries are not permitted. In order to make a customer order available for shipping, the requested items have to be collected from the warehouse. Each customer order is processed separately, i.e. it may not be merged (batched) with other customer orders.

Human operators (order pickers) walk or ride through the warehouse, retrieving the items from known storage locations. Picking is performed on (picker) tours through the warehouse, i.e. each order picker starts from the depot, visits the locations of the items to be collected and, afterwards, returns to the depot where he/she deposits the collected items. The distance which has to be covered for collecting all items of an order and, correspondingly, the time, which is required to do so, is dependent on the sequence according to which the order picker visits the locations. The determination of the sequence is part of the picker routing problem. Thus, a solution to the picker routing problem, e.g. obtained by application of so-called routing strategies (Roodbergen, 2001), gives the processing time of an order, i.e. the time which passes from the moment the order picker leaves the depot until the moment he/she returns to the depot. As customer orders are processed separately, the processing times can be computed in advance and assumed to be given.

The number of order pickers is limited. Thus, each order picker will have to process several orders in sequence. When all items of an order have been collected and forwarded to the depot by a picker, the order is considered as finalized and available for shipping. The point in time when an order is finalized will be denoted as the release date of an order. It is determined by its processing time and the sum of the processing times of the orders which have been processed before by the respective order picker. In other words, the release date of an order is dependent on how the orders are assigned to order pickers and how they are scheduled.

A fleet of homogeneous vehicles is based at the warehouse. The vehicles perform tours from the warehouse to the customer locations and back on which the requested items are delivered to the customers. Thus, for each vehicle tour, it has to be decided which customers should be served and in which sequence they should be visited. Each tour is started by loading an available vehicle with the items requested by the customers assigned to the respective tour. Only items from orders finalized for shipping may be loaded, and all items of an order must be loaded completely on the same vehicle. The customer locations are visited one after another and the respective requested items are unloaded. Service times have to be taken into account for the loading operations at the warehouse as well as for the unloading operations at the customer locations. The length of a tour (i.e. its duration) can then be defined as the sum of all service times required at the warehouse and at the customer locations visited, plus the travel times needed by the vehicle for moving from the warehouse to the first customer location, between the customer locations, and from the last customer location back to the warehouse. It is limited by a driving time constraint (Prescott-Gagnon et al., 2010). The point in time when a vehicle returns to the warehouse after having visited all customers of a tour will be denoted as the completion date of this tour.

Loading of a vehicle for a particular tour could be started as soon as picking of all orders which have been assigned to the tour has been finalized. However, the number of vehicles is limited, and vehicles may have to perform multiple tours. Loading of the orders for a particular tour, thus, may have to wait until the vehicle has returned from a previous tour. The start date of a tour is correspondingly defined as the maximum of the release dates of all orders assigned to the tour and the completion date of the tour previously performed by the vehicle.

Each customer order is characterized by a certain due date, which has been agreed on by the warehouse and the customer. The point in time when the requested items of a customer order are actually unloaded at the customer location will be named the delivery date of the order. An order which has not been received by the customer by the due date results in customer dissatisfaction, fines or even in the loss of the customer if such delays happen to occur over a longer period of time. Complying with agreed due dates, therefore, is of uttermost importance to the economic success of distribution warehouses and will provide the core criterion for the evaluation of how the warehouse manages to process customer orders. Since the consequences of delayed deliveries are often dependent on the length of the delays, we will refer to the total tardiness of all customer orders here (also see, e.g., Ullrich (2013) who have used the total tardiness as an evaluation criterion in similar problem settings). In case that the due date of an order is not met, its tardiness equals the difference between the delivery date and the due date. If the order is delivered in time, the tardiness of the order amounts to zero. Then, the total tardiness of a set of customer orders equals the sum of the tardiness of all orders in the set.

The OASVRP can now be stated. Let the following be given:

- a set of customers and their locations, a limited number of order pickers, and a homogeneous fleet of vehicles,
- a set of customer orders with agreed due dates and (picking) processing times,
- travel times between the customer locations and between the warehouse and the customer locations,
- a limit on the tour length,
- a service time required for loading the vehicles at the warehouse and a service time required for unloading the vehicles at the customer locations.

The following six questions have then to be answered (simultaneously) such that the total tardiness of all customer orders is minimized and the given limit on the tour length is not exceeded:

1) For each customer order, to which order picker should it be assigned?
2) For each order picker, in which sequence should the assigned customer orders be processed?
3) For each customer location, to which tour should it be assigned?
4) For each tour, in which sequence should the assigned customer locations be visited?
5) For each tour, to which vehicle should the tour be assigned?
6) For each vehicle, in which sequence should the tours assigned to the vehicle be processed?

Fig. 1 illustrates a solution of an instance of the OASVRP with nine customer orders, two pickers and two vehicles. Fig. 1a depicts the temporal aspects by means of a Gantt chart; it further demonstrates the assignment of customer orders to order pickers and the sequence according to which orders are processed, as well as the assignment of tours to vehicles and the sequence according to which the tours are performed. Picker #1 processes customer order #1 first, then continues with order #2, order #3 and order #5. Correspondingly, picker #2 starts with processing customer order #6, followed by orders #7, #4, #8, and #9. The (picking) processing time of each order is represented by the length of the corresponding rectangle. The right end of each rectangle provides the release date of the corresponding order.

Four tours have been built for delivering the requested items to the customers. Fig. 1b provides a graph of the corresponding routes. From Fig. 1a it can be taken that each vehicle executes two tours. E.g., vehicle #1 visits customer #1 first and then proceeds to customer #2 on a first tour; on a second tour customers #9 and #3 are visited. The length of each rectangle represents the time which is needed by the vehicle for traveling from the warehouse or a previous customer to the respective customer location plus the service time for unloading the vehicle at the customer location. The rectangle at the beginning of each tour indicates the service time for loading the vehicle at the warehouse, while a rectangle labeled with 0 represents a trip of the vehicle from the last customer of the tour back to the warehouse.

Fig. 1a also demonstrates that a tour cannot be started before all corresponding orders have been finalized at the warehouse. E.g., loading of vehicle #1 for the first tour (1, 2) starts as soon as picking of customer orders #1 and #2 has been completed. After having visited customer #2, vehicle #1 returns to the warehouse where it remains idle until picking of the last order of its second tour has been finalized. Loading of vehicle #1 for the second tour (9, 3) commences when order #9 has been provided. The first tour (7, 8, 6) of vehicle #2 cannot be started before picking of the last order (order #8) has been finalized. While this tour is being carried out, picking of all orders of the second tour (4, 5) is completed. Thus, loading of vehicle #2 can immediately be started upon its return to the warehouse from the first tour.

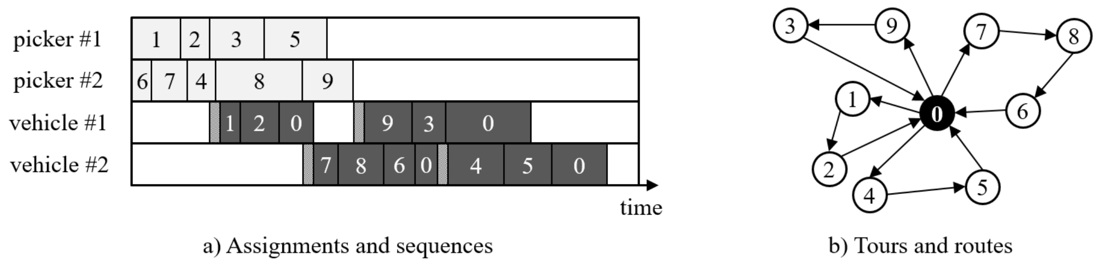


Fig. 1: Example solution

We note that the OASVRP is actually composed of two subproblems, namely an order assignment and sequencing problem (OASP), where customer orders have to be assigned to order pickers and sequences have to be determined in which the orders should be processed, and a vehicle routing problem (VRP), where the vehicles may perform multiple tours and release dates and due dates have to be considered. The interface between these two problems is provided by the release dates of the orders on the one hand and the start dates of the tours on the other hand. Before we introduce a solution approach to the OASVRP, we will review the literature related to the two subproblems and the OASVRP.

3 Literature review

3.1 Order assignment and sequencing problem

If we assume that customers are served individually and not on a tour together with other customers, then the time that it takes to transport the requested items from the depot to a customer is fixed and the latest release date of each customer order could be computed from the given due dates. This gives rise to the order assignment and sequencing problem which can be stated as follows: Let a set of customer orders with (latest) release dates and processing times, and a limited number of order pickers be given. How should the customer orders be assigned to the order pickers and in which sequence should the assigned customer orders be processed by each order picker such that the total tardiness of all orders is minimized?

To the best of our knowledge, the OASP has not been addressed in the context of order picking so far. However, the OASP is equivalent to the Identical Parallel Machine Problem (IPMP) from scheduling (Pinedo, 2016), where the customer orders are processed by machines instead of order pickers.

The IPMP is known to be NP-hard (Pinedo, 2016) and only a few exact solution approaches are available.

The first exact approach, a dynamic programming-based algorithm, was introduced by Gupta & Maykut (1973). Branch and bound algorithms were proposed by Azizoglu & Kirca (1998), Yalaoui & Chu (2002) and Shim & Kim (2007) which managed to solve problems with up to 15 customer orders and three machines, 20 customer orders and two machines, and 30 customer orders and five machines, respectively.

Due to the fact that exact algorithms can only deal with small-sized problems, heuristic approaches have been developed for solving problems of practical size. In particular, priority rule-based algorithms (Ho & Chang, 1991) have been introduced, where customer orders are sorted first according to a priority rule and then assigned one by one to the next available machine. The priority of a customer order can be dependent on its due date, on its processing time and/or on the ratio between the average processing time and the average due date. Alidaee & Rosa (1997) extended the modified-due-date (MDD) rule for the single machine problem proposed by Baker & Bertrand (1982). The MDD rule combines elements of the earliest-due-date (EDD) rule, where priority values of orders increase with a decreasing due date, and the shortest-processing-time rule, according to which higher priorities are assigned to orders with shorter processing times. Other approaches than priority-rule-based algorithms have been proposed by Koulamas (1997), who designed a decomposition approach as well as a hybrid simulated annealing algorithm. According to Ullrich (2013), the current state-of-the-art heuristic has been developed by Biskup et al. (2008). In this approach, customer orders are first sorted according to the EDD rule. Several incomplete initial solutions are then generated, where each initial solution is iteratively completed. In each iteration, exactly one non-assigned customer order is inserted into the partial solution. For each machine, one customer order is selected according to the MDD rule, resulting in a set of potentially assignable customer orders. One customer order from this set is then selected and optimally inserted into the partial solution.

3.2 Vehicle routing problems with multiple use of vehicles, release dates and due dates

The second subproblem of the OASVRP deals with the determination of routes for the vehicles which are used for delivering the requested items to the customers after finalized orders have been provided at the warehouse. It represents a variant of the classic vehicle routing problem where each customer order is characterized by a release date, i.e. a point in time when it becomes available for shipment at the depot, and a due date, i.e. the point in time by which it should have been received at the customer location. A set of homogeneous vehicles is available for transporting the requested items to the customers. Several customer locations may be visited on each tour and the vehicles may be used for multiple tours; however, the length of each tour is limited.

A VRP with multiple use of vehicles and a tour length constraint has been introduced by Fleischmann (1990). He extended the classic capacitated vehicle routing problem and designed a savings based heuristic. Taillard et al. (1996) and Brandão & Mercer (1997) proposed tabu search algorithms in order to solve the VRP with multiple use of vehicles. A constructive heuristic has been introduced by Petch & Salhi (2004), and Olivera & Viera (2007) suggested an extended tabu search approach to this problem.

For the VRP with hard time windows and multiple use of one vehicle, Azi et al. (2007) proposed an exact algorithm. Azi et al. (2010) extended this work to the case of multiple vehicles. They suggested a branch-and-price algorithm that is able to solve instances with up to 50 customers. More recently, Azi et al. (2014) designed an adaptive large neighborhood search algorithm for the heuristic solution of large-sized instances.

A VRP with release dates has been considered by Cattaruzza et al. (2016). They designed a genetic algorithm for the vehicle routing problem with hard time windows and release dates.

Due dates can be considered as a special case of time windows in which the lower bound is sufficiently small and the upper bound of the time window is a soft constraint. Taillard et al. (1997) suggested a tabu search algorithm for a VRP with a hard constraint regarding the lower bound and a soft constraint with respect to the upper bound of the time window. For the VRP with soft time windows, Chiang & Russell (2004) and Fu et al. (2008) designed tabu search algorithms. Liberatore et al. (2011) also considered the VRP with soft time windows and developed a branch-and-price algorithm.

3.3 Integrated scheduling and vehicle routing problems

As mentioned before, the OASP is equivalent to the IPMP. This scheduling problem has also been considered in conjunction with distribution problems which involve routing decisions. Table 1 provides an overview of publications related to such integrated scheduling and vehicle routing problems (ISVRP). (We refer to Chen (2010) for a very detailed review.) The second column of the table depicts the number of machines considered in the scheduling subproblem. The third, fourth and fifth column refer to the routing subproblem. The third column provides the number of available vehicles, while the entry "infinite" indicates that a sufficiently large number of vehicles has been assumed. The fourth column indicates whether each vehicle may only perform a single tour or whether it can be used for multiple tours, while the fifth column informs whether a limit on the length of each tour has been considered.

Table 1: Integrated scheduling and vehicle routing problems

reference	# machines	# vehicles	use of vehicles	tour length
Hurter & Van Buer (1996)	single	infinite	single	unlimited
Van Buer et al. (1999)	single	intinite	multiple	limited
Chen & Vairaktarakis (2005)	multiple	infinite	single	unlimited
Li et al. (2005)	single	single	multiple	unlimited
Low et al. (2013)	single	multiple	single	unlimited
Low et al. (2014)	single	infinite	single	unlimited
Li et al. (2016)	single	infinite	single	unlimited
Ullrich (2013)	multiple	multiple	multiple	unlimited
this paper (OASVRP)	multiple	multiple	multiple	limited

Hurter & Van Buer (1996) were the first who considered an ISVRP (Gao et al., 2015). They investigated a newspaper production and distribution problem. Different types of newspapers have to be delivered from a distribution center to drop-off points. All drop-off points have to be served by an identical deadline. The delivery of the newspapers is performed on tours by a fleet of homogeneous vehicles, but only one type of newspapers can be included in a single tour. Van Buer et al. (1999) extended this problem by allowing multiple tours per vehicle and multiple types of newspapers per tour. Chen & Vairaktarakis (2005) studied several variants of the ISVRP with an unrestricted number of homogeneous vehicles. Each vehicle is allowed to perform at most one tour and can visit a restricted number of locations per tour. In the production subproblem, both a single machine and multiple machines are assumed. Li et al. (2005) considered an ISVRP, where all customer orders are processed by a single machine. A single vehicle is available which may perform multiple tours. Low et al. (2013) investigated an ISVRP which integrates a scheduling problem with one machine and a VRP with hard time windows. This problem was extended by Low et al. (2014) who took a fleet of heterogeneous vehicles into account. Moreover, instead of hard time windows, soft time windows were assumed. The work of Li et al. (2016) combines a scheduling problem including a single machine with routing decisions including an unlimited number of homogeneous vehicles. In Ullrich (2013), multiple machines are used in the production process and a limited number of heterogeneous vehicles is used for the delivery of customer orders. For machines and vehicles, ready dates are given, i.e. the corresponding machine or vehicle must not necessarily be available at the beginning of the planning horizon.

As can be seen from Table 1, the problem considered by Ullrich (2013) resembles the OASVRP dealt with in this paper, except for the limitation of the tour lengths. The author proposed a genetic algorithm and investigated the benefits from an integrated solution of the scheduling and routing problems. However, the performance of the algorithm deteriorates drastically with an increasing number of customer orders. E.g., for instances with 70 orders, the quality of solutions provided by the genetic algorithm is hardly superior to the quality of solutions generated by a simple construction procedure (Ullrich, 2013, p. 163).

Apart from incorporating the tour length limitation, we will, therefore, pay particular attention to the development of an algorithm for the OASVRP which is capable of providing high-quality solutions to practical-sized problem instances in reasonable computing times. The algorithm, based on an iterated local search approach, will be presented in the next section.

4 Iterated local search approach

4.1 General principle

Iterated local search (ILS) has successfully been adapted to many kinds of optimization problems. It can be considered as the state-of-the-art algorithm for operations research problems, among others for various types of vehicle routing problems (Vidal et al., 2013), for single machine (Grosso et al., 2004; Congram et al., 2002) and identical parallel machine scheduling problems (Brucker et al., 1996, 1997) as well as for the order batching and sequencing problem (Henn & Schmid, 2013).

The general principle of ILS can be described as follows (Lourenço et al., 2010): Starting with an initial solution σ_{ini}, an improvement phase is executed in order to determine a local optimum, resulting in the first incumbent solution σ_{inc} and, at the same time, the best solution σ^* found so far. Perturbation and improvement phases are then alternately performed until a termination condition is met. In the perturbation phase, the incumbent solution σ_{inc} is randomly modified in order to avoid the ILS getting stuck in a local optimum. Based on the modified solution, a (new) local optimum is determined by means of the improvement phase. If the resulting solution represents a new best solution, then the best solution σ^* as well as the incumbent solution σ_{inc} are updated. Otherwise, σ^* remains unchanged and σ_{inc} is only altered if an acceptance condition is met. Depending on the acceptance condition, it may be possible to accept a solution with a worse objective function value than the incumbent solution. A pseudocode of the ILS approach is depicted below.

Algorithm 1 General principle of iterated local search

Input: problem data

Output: solution σ^* to the OASVRP and corresponding total tardiness $f(\sigma^*)$

generation of an initial solution σ_{ini}
σ_{inc} := improvement(σ_{ini})
$\sigma_{best} := \sigma_{inc}$
while termination condition is not met **do**
  $\tilde{\sigma}$:= perturbation(σ_{inc})
  σ^* := improvement($\tilde{\sigma}$)
  if $f(\sigma^*) < f(\sigma_{best})$ **then**
    $\sigma_{best} := \sigma^*$
  end if
  σ_{inc} := acceptance condition(σ^*, σ_{inc})
end while

4.2 Initial solution

For the generation of an initial solution, the OASVRP is divided into its two subproblems, which are then solved sequentially. First, a solution to the VRP is constructed. Tours and corresponding routes are generated by adapting the EDD rule originally designed for the IPMP (Baker & Bertrand, 1981). According to this rule, all customer orders are sorted in a non-descending order of the due dates. Then, in this sequence, the orders are assigned to the vehicle which currently possesses the shortest total travel time. More precisely, a customer order is assigned to the last position of the currently last route of the vehicle chosen. A new tour is opened each time the maximum tour length would be exceeded. In order to provide a feasible solution to the VRP, order release dates have to be taken into account. Regarding the OASVRP, the release date of an order is defined by the point in time when the order is finalized for shipment at the warehouse, which is not known at this stage of the algorithm. Therefore, release dates are estimated by assuming that the number of order pickers is identical to the number of vehicles and each order picker processes all orders assigned to a certain vehicle in the sequence provided by the above-described modification of the EDD rule. Based on the estimated release dates, the start date of each tour is determined. The estimated release dates are taken as (planned) start dates of the tours.

The solution of the VRP is then taken as input for the solution of the OASP which determines the release dates of the orders. Regarding the OASVRP, picking of an order has to be finalized before the start date of the tour in which it is included. In the context of the resulting OASP, the tardiness of an order is then defined as the non-negative difference between the (planned) start date of the corresponding tour and the release date of the order. As mentioned in Section 3.1, the OASP is equivalent to the IPMP. Thus, in order to solve this problem, the approach of Biskup et al. (2008) is applied.

If the approach of Biskup et al. (2008) leads to a solution with a total tardiness equal to 0, combining this solution with the solution to the VRP results in a feasible solution to the OASVRP as well. Otherwise, a tour including orders for which picking is completed after the (planned) start date of the tour has to be postponed and the start dates of the respective tour and the subsequent tours are corrected correspondingly. After having obtained a feasible solution, start dates may be updated in order to ensure that each tour starts as early as possible.

4.3 Improvement phase

As has been explained in Section 2, solving the OASVRP involves six different types of decisions which have to be taken simultaneously. Since a simple local search procedure will not be able to deal with all aspects of the problem, a more complex improvement procedure will be used. In fact, a variable neighborhood descent (VND) algorithm has been designed in order to tackle all decision types.

VND was first introduced in Hansen & Mladenovic (2001). In this approach, the solution space is explored using a sequence of neighborhood structures $\mathcal{N}_1, \ldots, \mathcal{N}_L$. Starting with a solution σ, a local optimum regarding the first neighborhood structure $\mathcal{N}_1$ is determined. If the resulting solution provides a better objective function value than the best solution found so far, this solution becomes the new best solution and $\mathcal{N}_1$ is explored again. Otherwise, the algorithm continues with exploring the next neighborhood structure. Each time a local optimum represents a new best solution, the algorithm continues with $\mathcal{N}_1$. The VND approach terminates when no improvement has been found in the last neighborhood structure $\mathcal{N}_L$. In this case, the best solution σ^* is a local optimum with respect to all neighborhood structures.

The improvement phase of the ILS approach presented in this paper consists of two VND algorithms, dealing with decisions related to the VRP (VND_VRP) and the OASP (VND_OASP), respectively. A pseudocode of the improvement phase is given in Algorithm 2.

Algorithm 2 Improvement phase

Input: problem data, solution σ with objective function value $f(\sigma)$

Output: local optimum σ^*, $f(\sigma^*)$

```
do
    σ* := σ
    σ := VND_VRP (σ)
    σ := VND_OASP (σ)
while f(σ) < f(σ*)
```

Each VND algorithm deals with one subproblem only. This approach is chosen because of the structure of the OASVRP. Decisions regarding the VRP only affect this subproblem. Moves related to these decisions can be performed quite easily. In contrast to that, any changes regarding the OASP have an impact on the release dates of the orders and, therefore, such moves will also affect the start dates of the tours. Since moves regarding the OASP affect both subproblems, their execution is much more time-consuming. As a consequence, a VND approach is first applied to the VRP and modifications to solutions of the OASP are not carried out before a local optimum regarding the VRP has been found.

In the VND algorithm for the VRP, four neighborhood structures $\mathcal{N}_1^{\text{VRP}}, \ldots, \mathcal{N}_4^{\text{VRP}}$ are contained:

$\mathcal{N}_1^{\text{VRP}}$: a consecutive sequence of orders is moved to another position of the same tour;
$\mathcal{N}_2^{\text{VRP}}$: two tours assigned to different vehicles are exchanged;
$\mathcal{N}_3^{\text{VRP}}$: a consecutive sequence of orders is moved to a tour assigned to another vehicle;
$\mathcal{N}_4^{\text{VRP}}$: a consecutive sequence of up to two orders is removed from a tour and it is assigned to the same vehicle building a new tour.

The impact of moves performed in the VND_VRP procedure is exemplified for the neighborhood structure $\mathcal{N}_4^{\text{VRP}}$ (see Fig. 2). Regarding $\mathcal{N}_4^{\text{VRP}}$, a neighbor solution is constructed by first choosing a tour assigned to a certain vehicle. A consecutive sequence of orders is then removed from the tour, i.e. the tour is divided into two subsets. The first subset contains all orders which are still included in the tour. No further changes will be performed to this tour. The second subset consisting of the removed orders will form a new tour and will be assigned to another position of the same vehicle. The position is chosen in such a way that the total tardiness is minimized.

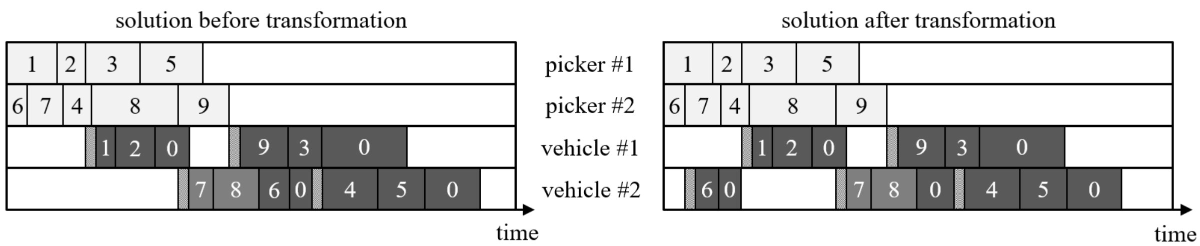


Fig. 2: Example of a move regarding neighborhood structure $\mathcal{N}_4^{\text{VRP}}$

In the example depicted in Fig. 2, a possible move regarding $\mathcal{N}_4^{\text{VRP}}$ is performed. The orders #7 and #8 are removed from the tour $(7, 8, 6)$ originally assigned to vehicle #2. The remaining tour only contains order #6 and can now start much earlier as order #6 is processed by an order picker at the very beginning of the planning horizon. The new tour $(7, 8)$ is inserted as the second one for vehicle #2 and can be started after order #8 has been provided. As can be seen from Fig. 2, the start date of this tour is identical to the start date of the tour $(7, 8, 6)$ in the solution before the transformation. Obviously, the length of the tour $(7, 8)$ is shorter than the length of the tour $(7, 8, 6)$. Therefore, the subsequent tour $(4, 5)$ can be started earlier now, resulting in earlier delivery dates for both orders.

The moves included in the VND_VRP procedure affect the tours assigned to one ($\mathcal{N}_1^{\text{VRP}}$ and $\mathcal{N}_4^{\text{VRP}}$) or two ($\mathcal{N}_2^{\text{VRP}}$ and $\mathcal{N}_3^{\text{VRP}}$) vehicles. Tours assigned to other vehicles will remain unchanged, but also decisions related to the OASP will not be affected. As mentioned before, moves changing the assignment of customer orders to order pickers or the sequence according to which customer orders are processed by an order picker are much more complex. Therefore, the VND_OASP procedure includes the following two neighborhood structures only:

$\mathcal{N}_1^{OASP}$: an order is moved to another position of the same picker;
$\mathcal{N}_2^{OASP}$: two orders assigned to different pickers are exchanged.

In Fig. 3, an example of a move regarding $\mathcal{N}_2^{OASP}$ is depicted, where the assignment to the order pickers is exchanged for orders #1 and #8. The release date of order #1 increases, resulting in a later start date of the corresponding tour. The start date of the following tour $(9, 3)$ is now determined by the completion date of the tour $(1, 2)$ instead of the release date of order #9. Consequently, the tour $(9, 3)$ is also postponed. Regarding vehicle #2, it can be seen that the start dates of the tours significantly decrease due to the exchange of orders #1 and #8 because the release date of order #8 decreases.

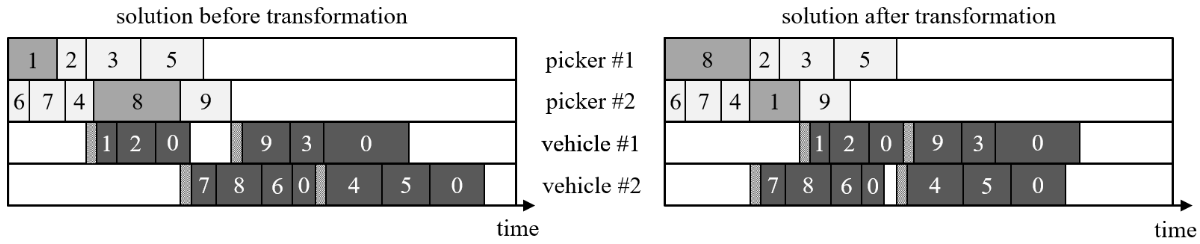


Fig. 3: Example of a move regarding neighborhood structure $\mathcal{N}_2^{\text{OASP}}$

4.4 Perturbation phase

After a local optimum has been identified in the improvement phase, the solution is randomly modified in the perturbation phase. The design of the perturbation phase is pivotal for the performance of an ILS approach. If the modifications are too small, a further application of the improvement phase will result in the same local optimum. If too many changes are applied to the local optimum, the promising part of the solution space is left and the ILS algorithm turns into an improvement procedure with multiple random starts (Lourenço et al., 2010).

For the perturbation phase, we decided to use moves related to the OASP, as their impact on the solution is expected to be larger than that of VRP moves (see the previous subsection). A move in the perturbation phase is defined by the exchange of two sequences of consecutive customer orders which are assigned to different order pickers (see Fig. 4). The lengths of the two sequences are chosen randomly and may be different from each other. The maximum length of a sequence determines the degree of modification performed in the perturbation phase.

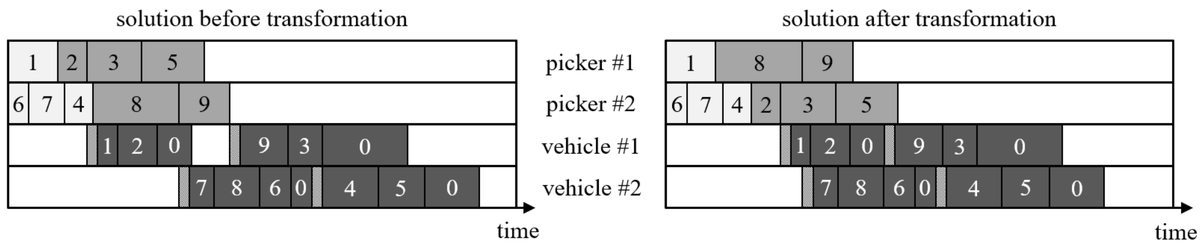


Fig. 4: Example of a move in the perturbation phase

In Fig. 4, three consecutive orders assigned to picker #1 are exchanged with two orders processed by picker #2. Consequently, the release dates change for all of the five orders. This affects the start date of all tours. Regarding vehicle #1, the first tour is postponed because of the increased release date of order #2. Nevertheless, the subsequent tour $(9, 3)$ is started earlier now. For vehicle #2, the transformation leads to earlier start dates for both tours. The start date of tour $(7, 8, 6)$ decreases due to the smaller release date of order #8, and the subsequent tour can then be started earlier as well since its start date is determined by the completion date of tour $(7, 8, 6)$.

4.5 Acceptance and termination condition

In addition to the perturbation phase, an adequate acceptance condition helps to overcome local optima. A solution of inferior quality may be accepted if no improvements have been found for a certain number $\tilde{n}$ of consecutive iterations and if the solution provides an objective function value not too far from the objective function of the current best solution. The acceptance condition of our ILS was proposed by Dueck & Scheuer (1990). According to this condition, a solution σ is accepted if its objective function value $f(\sigma)$ is not larger than $(1 + \alpha) \cdot f(\sigma^*)$, where $f(\sigma^*)$ denotes the objective function value of the current best solution σ^* and α is a parameter indicating the relative amount of deterioration allowed. At the beginning of the algorithm, α is initialized by 0, i.e. a solution is only accepted if it represents an improvement compared to the current best solution. Each time no solution is accepted for $\tilde{n}$ consecutive iterations, α is increased. Whenever the incumbent solution is updated, α is set to 0 again. Thus, the longer the ILS algorithm gets stuck in a local optimum the higher the relative amount of deterioration allowed gets. This type of acceptance condition has also been applied by Polacek et al. (2004) to a VRP with time windows and multiple depots, by Tarantilis et al. (2004) to a VRP with a heterogeneous fleet and by Henke et al. (2015) to a VRP with multiple compartments.

A time limit has been chosen as the termination condition. After each iteration, it is checked whether the time limit has been exceeded. If this is the case, the ILS approach is terminated. Otherwise, at least one additional iteration is performed.

5 Numerical experiments

5.1 Test problem instances and parameter settings

In order to evaluate the performance of the ILS algorithm as well as to determine the benefits from solving the OASVRP as a holistic problem instead of dealing with the OASP and the VRP sequentially, extensive numerical experiments have been conducted. Since the problem instances of Ullrich (2013) were not available, new test problem instances have been generated. The generation of the instances followed the procedures of Ullrich (2013) for the VRP and of Scholz et al. (2016) for the OASP.

For the numerical experiments, problem instances with 100 and 200 customer orders have been generated. Instances of this size have also been used by Henn (2015) and Scholz et al. (2016) for different types of order picking problems and they are also considered as realistic problem sizes for VRPs (Desaulniers et al., 2014). The customer orders are processed by 2, 3 or 5 order pickers in the warehouse. For the determination of the processing times of the customer orders, a block layout with 10 picking aisles is assumed for the picking area. A class-based procedure has been assumed for assigning the articles to the storage locations (Henn, 2015; Scholz et al., 2016). The routes of the order pickers are constructed by means of the S-shape strategy which represents the routing strategy most frequently used in practice (Roodbergen, 2001). Processing times of orders will increase with an increasing number of blocks. Instances with 1 block (short processing times) and 3 blocks (long processing times) are considered.

Identical to Ullrich (2013), the number of vehicles available for the delivery of the customer orders is set either to 4, 6, 8 or 10. The respective customer locations are chosen randomly, while the corresponding coordinates for the horizontal and vertical dimensions are selected from the interval $[1, 100]$ for instances including 100 customer orders and from $[1, 150]$ for problems with 200 orders. The location of the warehouse is fixed to the coordinates $(50, 50)$ and $(75, 75)$, respectively. The travel times are then defined by the euclidean distances between the locations (Ullrich, 2013). The time for loading the vehicle (service time at the depot) is set to 20 minutes, while 5 minutes are required for unloading the required items (service time at a customer location). The maximum tour length is set to 8 hours.

Finally, a due date is assigned to each customer order. The due dates are determined based on the procedure of Ullrich (2013). According to this procedure, the due dates are dependent on the number of customer orders N, the number of vehicles K, the number of order pickers M as well as on the processing times p_n $(n = 1, \ldots, N)$ of the orders and the travel and service times. Additionally, a parameter θ is introduced describing how difficult it is to meet the due dates. The due date of customer order n is then a realization of the random variable D_n which is defined as follows (Ullrich, 2013):

$$D_n = p_n + t_{0n} + s_0 + s_n + \Gamma + \Delta \tag{1}$$

On the one hand, D_n includes order-specific data such as the travel time t_{0n} between the depot and the location of customer n and the service time (s_n) at the customer location. Due to the integration of the service time at the depot (s_0) as well as the random variables Γ and Δ, general problem data is included in the calculation on the other hand. Γ and Δ are uniformly distributed over the discrete sets $\{0, \ldots, \lfloor \theta \, (\max_{n=1,\ldots,N} p_n) \, (N/ \, (K + M)) \rfloor\}$ and $\{0, \ldots, \lfloor \theta \, (\max_{n=1,\ldots,N} p_n) \rfloor\}$, respectively. In the numerical experiments, θ is set to 0.5 (tight due dates) and to 1.0 (loose due dates).

The combination of all parameter values gives rise to 96 different problem classes. For each class, 48 test problem instances have been generated, resulting in 4608 instances in total. The ILS algorithm has been implemented using Visual Studio C++ 2015. The numerical experiments have been performed by means of a Haswell system with up to 3.2 GHz and 16 GB RAM per core.

Regarding the ILS approach, the following settings have been chosen. The maximum length of a sequence of consecutive orders exchanged is set to 5 for the perturbation phase. The parameter α included in the acceptance criterion is increased by 0.1 after 50 consecutive iterations each without finding a new best solution. The time limit for the ILS has been fixed to 30 minutes for instances with 100 customer orders and to 60 minutes for problems with 200 orders.

5.2 Generation of upper bounds

As has been shown by Ullrich (2013), only very small problem instances can be solved to optimality within a reasonable amount of computing time. Therefore, upper bounds are generated in order to evaluate the performance of the ILS approach. In fact, three procedures for the determination of upper bounds are applied.

As for the first procedure (Ullrich, 2013), each customer order is assumed to be served on a separate tour. This assumption reduces the OASVRP to a hybrid flow shop problem with M parallel machines at the first stage and K parallel machines at the second stage. Processing times at the first stage are given by the processing times p_n ($n = 1, \ldots, n$) of the orders, while the times for delivering the customer orders (given by $s_0 + t_{0n} + t_{n0} + s_n$) represent the processing times at the second stage (Ullrich, 2013). This problem is then solved by applying the MDD rule which has been proven to perform quite well for multiple stage hybrid flow shop problems with due dates (Brah, 1996). Ullrich (2013) compared the solutions generated by a genetic algorithm to this upper bound and pointed out that the genetic algorithm was not able to find solutions of superior quality for problem instances with 70 or more customer orders. This observation indicates that the genetic algorithm does not perform well for those problems. Therefore, in our experiments, this upper bound (UB_1) is used in order to identify whether the ILS approach is suitable for dealing even with very large instances.

The general principle for the generation of the second upper bound (UB_2) was also proposed by Ullrich (2013). He suggested to divide the problem into its two subproblems and then solve them one after another to optimality. The author started with the VRP, continued with the OASP and got back again to the VRP. The procedure then terminates since performing further iterations have proven not to lead to significant improvements regarding the solution quality. Ullrich (2013) computed upper bounds of this type for very small problem instances including 7 customer orders only. Therefore, in order to be able to calculate the bounds for larger instances, we use the same principle but the subproblems are solved heuristically. At first, the procedure for the determination of an initial solution (see Section 4.2) is used, i.e. the VRP is solved and then the algorithm of Biskup et al. (2008) is applied to the OASP. As suggested by Ullrich (2013), the VRP is then solved again. Here, the VND_VRP procedure (see Section 4.3) is applied. This procedure for the generation of an upper bound is much more complex than the previous one, as a state-of-the-art algorithm is used for solving the OASP and a VND approach for solving the VRP. Upper bounds of this second type are generated in order to determine the benefits from dealing with the OASVRP as a holistic problem instead of solving the subproblems in sequence.

The determination of the third upper bound (UB_3) also originates from the ILS approach. The initial solution is constructed and one improvement phase of the ILS algorithm is performed. The quality of this bound is at least as good as the quality of the second bound. The third bound is used for the investigation of the impact of the perturbation phase on the quality of solutions provided by the ILS approach.

5.3 Evaluation of the solution quality of the ILS algorithm

In Tables 2 and 3, the average total tardiness (tard_i) in minutes is depicted for the upper bounds UB_i ($i \in \{1, 2, 3\}$) as well as for the solutions provided by the ILS approach (tard_{ILS}) for problem classes with 100 and 200 customer orders, respectively. Furthermore, the average improvements (imp_i) [in %] are presented in comparison to upper bound UB_i. In the tables, K denotes the number of vehicles, B represents the number of blocks and θ is the parameter used for the generation of the due dates (see Section 5.1).

Performance of the ILS algorithm for large-sized instances

Comparing the objective function values of solutions obtained by the ILS approach to the upper bound UB_1, significant improvements regarding the total tardiness can be observed. On average, the reduction ranges from 4.6% (100 orders, 2 pickers, $\theta = 0.5$, $B = 1$, $K = 10$) to 94.0% (200 orders, 5 pickers, $\theta = 1$, $B = 3$, $K = 4$). The magnitude of the improvement varies very strongly between different problem classes. This can be explained by the performance of the approach for generating UB_1. Application of the MDD rule leads to rather good solutions to the OASP. The VRP is solved on the basis of the assumption that each customer is served on a separate tour. This assumption is not critical as long as processing the orders in the warehouse consumes more time than the separate delivery of each order, i.e. when many more vehicles than order pickers are available or when the processing times of the orders are large in comparison to their travel times. Furthermore, the upper bound may have a good quality in case of loose due dates.

Due to the increasing ratio between vehicles and order pickers, the amount of improvement decreases with an increasing number of vehicles (75.4% for $K = 4$ and 32.6% for $K = 10$) and increases with an increasing number of order pickers (30.6% for 2 pickers and 75.6% for 5 pickers). As has been anticipated, these two parameters have the largest impact on the amount of improvement. If few vehicles are available for the delivery, the solutions to the VRP can significantly be improved by serving several customer orders on the same tour when the orders can be processed by many order pickers. Besides the number of pickers and the number of vehicles, the number of blocks and the parameter θ affect the amount of improvement. An increasing number of blocks results in a reduction of the average improvement. While the total tardiness can be reduced by 56.7% in a single-block layout, the improvement amounts to 47.5% when the picking area is composed of three blocks. The reason can be found in the processing times which increase when the picking area includes a larger number of blocks. Picking the orders gets more time-consuming and the advantage of serving several customers on a single vehicle tour diminishes. The amount of improvement also decreases with an increasing value of θ, as the bound can be improved by 55.5% for $\theta = 0.5$ and by 47.7% for $\theta = 1$. Problem classes characterized by a large θ contain instances with loose due dates. In this case, delivering more than one order per tour is not that important. Thus, the quality of the bound increases and the amount of improvement obtained by application of the ILS approach decreases. Furthermore, it can be observed that larger reductions of the total tardiness are achieved for instances with a larger number of orders (43.8% for 100 orders and 60.4% for 200 orders).

Table 1: Evaluation of the ILS approach for problem classes with 100 customer orders

			2 pickers							3 pickers							5 pickers						
θ	B	K	$tard_1$	imp_1	$tard_2$	imp_2	$tard_3$	imp_3	$tard_{ILSA}$	$tard_1$	imp_1	$tard_2$	imp_2	$tard_3$	imp_3	$tard_{ILSA}$	$tard_1$	imp_1	$tard_2$	imp_2	$tard_3$	imp_3	$tard_{ILSA}$
1	1	4	1111	61.9	593	28.7	549	23.0	423	1139	79.8	528	56.5	449	48.8	230	1196	91.2	576	81.8	495	78.8	105
1	1	6	707	30.6	605	19.0	572	14.3	490	691	60.9	381	29.2	343	21.2	270	724	84.5	317	64.5	259	56.5	112
1	1	8	565	6.9	643	18.1	605	13.0	526	488	39.2	390	23.9	359	17.4	297	500	74.1	262	50.5	216	40.0	130
1	1	10	576	5.4	643	15.2	609	10.5	545	397	21.7	395	21.4	362	14.2	311	373	63.4	231	40.8	195	29.9	136
1	3	4	901	47.0	647	26.1	616	22.4	478	938	75.5	482	52.3	416	44.7	230	1029	93.3	564	87.8	477	85.5	69
1	3	6	657	10.9	729	19.7	694	15.6	586	551	44.6	430	29.1	400	23.6	305	597	81.5	294	62.4	239	53.7	111
1	3	8	714	10.7	773	17.5	736	13.4	638	407	15.2	458	24.5	428	19.3	345	391	65.7	271	50.5	227	40.9	134
1	3	10	774	11.7	796	14.1	767	11.0	683	423	9.9	492	22.5	459	16.9	381	286	48.9	259	43.6	222	34.3	146
0.5	1	4	1276	54.3	772	24.5	722	19.3	583	1261	71.4	682	47.1	602	40.0	361	1238	84.1	675	70.8	598	67.0	197
0.5	1	6	823	25.2	745	17.3	709	13.2	616	793	51.9	503	24.2	464	17.9	381	784	75.0	405	51.5	345	43.0	196
0.5	1	8	665	6.6	742	16.3	704	11.7	622	584	34.1	485	20.6	451	14.7	385	555	63.6	340	40.7	291	30.5	202
0.5	1	10	660	4.6	726	13.2	694	9.3	630	471	16.8	483	18.9	451	13.0	392	415	51.8	298	33.1	258	22.6	200
0.5	3	4	1193	38.5	942	22.0	911	19.4	734	1177	62.8	737	40.6	665	34.2	438	1193	81.2	752	70.2	667	66.3	224
0.5	3	6	879	10.5	961	18.2	923	14.8	786	741	36.0	623	23.8	590	19.6	474	735	67.7	448	46.9	386	38.4	238
0.5	3	8	901	9.7	957	15.0	921	11.7	813	572	12.6	628	20.3	593	15.7	500	521	51.5	405	37.6	360	29.8	253
0.5	3	10	920	10.7	943	12.9	908	9.6	821	556	9.1	620	18.4	586	13.8	506	400	35.7	379	32.2	339	24.1	257
average			833	21.6	763	18.6	728	14.5	623	699	40.1	520	29.6	476	23.4	363	684	69.6	405	51.4	348	46.3	169

Table 2: Evaluation of the ILSA for problem classes with 200 customer orders

			2 pickers							3 pickers							5 pickers						
θ	B	K	$tard_1$	imp_1	$tard_2$	imp_2	$tard_3$	imp_3	$tard_{ILSA}$	$tard_1$	imp_1	$tard_2$	imp_2	$tard_3$	imp_3	$tard_{ILSA}$	$tard_1$	imp_1	$tard_2$	imp_2	$tard_3$	imp_3	$tard_{ILSA}$
1	1	4	7742	76.5	4388	58.5	4222	56.9	1820	7853	86.8	4497	77.0	4351	76.2	1034	8035	92.6	4670	87.3	4511	86.9	591
1	1	6	4938	58.5	2715	24.5	2543	19.4	2050	5013	76.6	2643	55.6	2386	50.8	1173	5169	88.6	2663	77.9	2383	75.3	589
1	1	8	3701	39.6	2493	10.4	2422	7.7	2234	3667	64.8	1980	34.7	1819	29.0	1293	3637	82.9	1941	67.9	1770	64.8	623
1	1	10	2929	21.0	2578	10.2	2482	6.7	2315	2846	51.4	1668	17.1	1577	12.4	1382	2858	76.7	1498	55.6	1333	51.1	665
1	3	4	6772	69.8	3391	39.9	3026	32.6	2039	7070	85.5	3529	70.9	3049	66.3	1027	7394	94.0	3909	88.7	3406	87.1	440
1	3	6	4336	42.3	2920	14.3	2847	12.1	2503	4400	70.2	2349	44.2	2114	38.0	1311	4590	88.4	2528	78.9	2286	76.7	533
1	3	8	3243	16.7	3037	11.0	2950	8.4	2702	3150	50.6	1867	16.7	1797	13.5	1555	3227	80.3	1689	62.4	1489	57.4	635
1	3	10	3273	12.0	3166	9.0	3072	6.2	2880	2458	31.6	1935	13.2	1855	9.4	1680	2478	71.0	1290	44.3	1135	36.7	718
0.5	1	4	8242	70.6	4950	51.0	4772	49.2	2425	8294	81.4	4948	68.8	4785	67.8	1541	8284	88.6	4967	80.9	4798	80.3	947
0.5	1	6	5381	52.6	3247	21.4	3070	16.9	2552	5341	70.0	3056	47.6	2803	42.9	1600	5342	83.0	2991	69.7	2697	66.2	907
0.5	1	8	3990	34.5	2889	9.5	2816	7.1	2615	3910	57.8	2318	28.8	2151	23.3	1651	3827	76.3	2203	58.9	2014	55.0	906
0.5	1	10	3236	18.1	2924	9.3	2830	6.3	2651	3072	44.9	1965	13.9	1876	9.9	1691	2979	69.3	1735	47.2	1565	41.5	915
0.5	3	4	7940	61.4	4557	32.7	4173	26.5	3066	8068	76.3	4554	58.0	4100	53.4	1912	8200	86.6	4773	77.0	4272	74.3	1098
0.5	3	6	5168	35.3	3828	12.7	3750	10.9	3343	5140	60.6	3167	36.1	2935	31.0	2024	5239	79.5	3171	66.2	2923	63.2	1072
0.5	3	8	3950	13.8	3778	9.9	3687	7.7	3404	3774	43.6	2512	15.2	2433	12.5	2130	3756	70.3	2239	50.2	2017	44.8	1114
0.5	3	10	3911	10.9	3819	8.7	3718	6.2	3485	2984	26.6	2503	12.5	2418	9.4	2191	2897	60.0	1756	34.0	1600	27.6	1158
average			4922	39.6	3417	20.8	3274	17.6	2630	4815	61.2	2843	38.2	2653	34.1	1575	4869	80.5	2751	65.5	2513	61.8	807

This can be explained by the fact that no advantage of the larger solution space is taken by the procedure for the construction of the upper bound, whereas the number of moves performed in the improvement phase of the ILS algorithm significantly increases with an increasing number of customer orders.

On average, across all problem classes, the first upper bound can be improved by 52.1%, which demonstrates that significant reductions of the total tardiness are obtained. In contrast to the genetic algorithm of Ullrich (2013), which is not able to significantly improve the upper bound for instances including 70 or more customer orders, the proposed ILS algorithm results in serious improvements even for very large instances. Furthermore, the impact of the parameters on the amount of improvement matches the expectations based on the quality of the upper bound, which leads us to the conclusion that the ILS algorithm is well designed.

Benefits of a holistic solution of the OASVRP

While the generation of the first upper bound makes use of two simple construction procedures, the second bound is provided by application of a state-of-the-art algorithm to the OASP and a VND approach to the VRP. Nevertheless, compared to the second upper bound, the ILS algorithm results in remarkable improvements, which vary between 8.7% (200 orders, 2 pickers, $\theta = 0.5$, $B = 3$, $K = 10$) and 88.7% (200 orders, 5 pickers, $\theta = 1$, $B = 3$, $K = 4$). Over all problem classes, the total tardiness can be reduced by 37.8% on average, which clearly demonstrates that solving the OASVRP as a holistic problem is pivotal for obtaining high-quality solutions.

The results from the experiments indicate that a simultaneous solution of the OASP and the VRP is more advantageous if the number of vehicles is not too large in comparison to the number of order pickers. If few vehicles are available for the delivery of the orders, more orders will be contained in a single tour. The start dates of the tours are then dependent on the release dates of several orders, i.e. the solution of the VRP is strongly affected by the solution of the OASP and the other way round. If the number of vehicles is very large, the tours include few or even a single order only. In this case, the VRP gets less important and the OASP can be solved without taking the vehicle tours into account. A similar argumentation holds for the impact of the number of blocks on the size of the improvement. Increasing processing times caused by a larger number of blocks produces the same effect as a decreasing number of order pickers does since fewer orders can be processed within the same amount of time. Thus, fewer customers will be visited on a tour, decreasing the benefits from solving the subproblems simultaneously. Regarding the number of customer orders, the results show that – what concerns the joint approach – a larger number of orders provides more space for improvement, increasing the reduction of the total tardiness by 8.3 percentage points (33.2% reduction for 100 orders and 41.5% reduction for 200 orders).

Impact of the perturbation phase on the solution quality

The third upper bound is obtained by application of the improvement phase to the solution provided by the procedure for the generation of the second bound. The improvements obtained by the ILS approach range between 6.2% (200 orders, 2 pickers, $\theta \in \{0.5, 1\}$, $B = 3$, $K = 10$) and 87.1% (200 orders, 5 pickers,

$\theta = 1$, $B = 3$, $K = 4$). On average, the improvement amounts to 32.5%, clearly demonstrating that more than a simple improvement procedure is required for providing high-quality solutions to such a complex problem. The application of the perturbation phase is pivotal in order to overcome local optima and to guide the search to the promising part of the solution space.

The amount of improvement is mainly dependent on the number of order pickers and the number of vehicles. Regarding the number of order pickers, it can be observed that the reduction of the total tardiness, given by UB_3, gets much larger with an increasing number of pickers. In fact, the average amount of reduction equals to 16.1% in the case of 2 pickers, while the total tardiness can be reduced by 53.1% for problems including 5 pickers. This can be explained by the fact that the perturbation phase exchanges randomly-chosen orders between two pickers. If 2 pickers are available only, the selection of the order pickers is fixed. Thus, the probability that the perturbation phase leads to solutions already investigated earlier is significantly increased in this case. Concerning the number of vehicles, the same behavior can be observed as for the comparison with UB_2: If the number of vehicles gets very large in comparison to the number of order pickers, many orders can be delivered on a separate tour. Thus, the benefit from solving the OASP and the VRP simultaneously diminishes, which also reduces the range in which improvements can be obtained. The impact of the processing times and the parameter θ is much less significant than the impact of the number of pickers and the number of vehicles. The amount of improvement obtained by application of the ILS approach increases with decreasing processing times, i.e. with a decreasing number of blocks, and an increasing value of θ. Furthermore, larger improvements are obtained for instances with a larger number of customer orders. While the total tardiness can be reduced by 28.1% when 100 customers are considered, an average reduction of 37.8% is obtained for instances with 200 orders. However, these results have to be taken carefully since much more computing time is spent on solving instances including 200 orders by application of the ILS approach.

5.4 Considerations regarding computing times

The generation of the upper bounds requires a few seconds of computing time, whereas the computing time of the ILS approach has been fixed to one hour for problem instances with 200 customer orders. The improvements obtained in comparison to the bounds provides information on the benefits of applying the ILS algorithm instead of using simple construction procedures or sequential solution approaches. However, no reliable conclusions on the performance of the proposed ILS algorithm can be drawn from the results. In particular, it is not known whether the time limit has appropriately been chosen. In Fig. 5, information on the development of the average solution quality over time is given for instances from three problem classes. Problem classes with 200 customer orders are considered, implying that the ILS approach is terminated after one hour of computing time. For each point in time, Fig. 5 depicts the relative deviation [in %] of the total tardiness provided by the best solution found after one hour from the tardiness of the current best solution.

The first problem class (2 pickers, $\theta = 0.5$, $B = 3$, $K = 10$), which has been considered, is characterized by very low improvements (6.2%) with respect to UB_3. After 10% of the total computing time, the tardiness provided by the current best solution can only be improved by 2% on average within the remaining 90% of the computing time. The reason can be found in the design of the perturbation phase. As mentioned before, fewer decisions have to be taken in the perturbation phase in the case of 2 pickers, which significantly reduces the number of possible moves. For the second problem class (3 pickers, $\theta = 1$, $B = 3$, $K = 6$), UB_3 can be improved by 38.0%, which can be interpreted as a fairly average amount of improvement. In this case, the time limit of one hour seems to be chosen appropriately. The total tardiness is reduced by 10% in the last 90% of the computing time, representing a significant amount of improvement. Thus, the algorithm should not be stopped at an early stage. Furthermore, the algorithm seems to converge as the reduction found amounts to 1.1% for the last 50% and 0.1% for the last 10% of the total computing time. The largest improvements are obtained for the problem class (5 pickers, $\theta = 1$, $B = 3$, $K = 4$), where UB_3 is reduced by 87.1% on average. The development of the solution quality over time clearly indicates that a larger amount of computing time is required for generating high-quality solutions. Even the tardiness obtained after 50% of the total computing time can be reduced by 11.1%. In the last 10% of the time, the objective function value can still be improved by almost 1%.

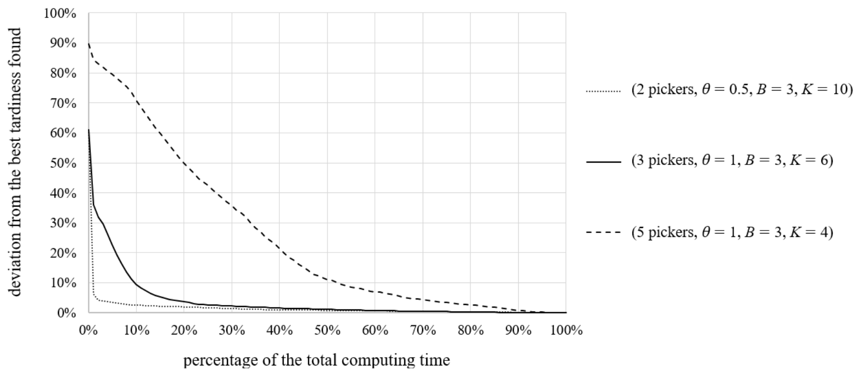


Fig. 5: Development of the solution quality over one hour of computing time

As a preliminary conclusion, it can be stated that the time limit of 1 hour is more than sufficient for solving instances from the class with 2 order pickers. For the classes considered above, which include 3 or 5 pickers, it is further investigated whether the computing time has been chosen sufficiently or not. Therefore, for these two problem classes, the development of the solution quality over four hours of computing time is depicted in Fig. 6.

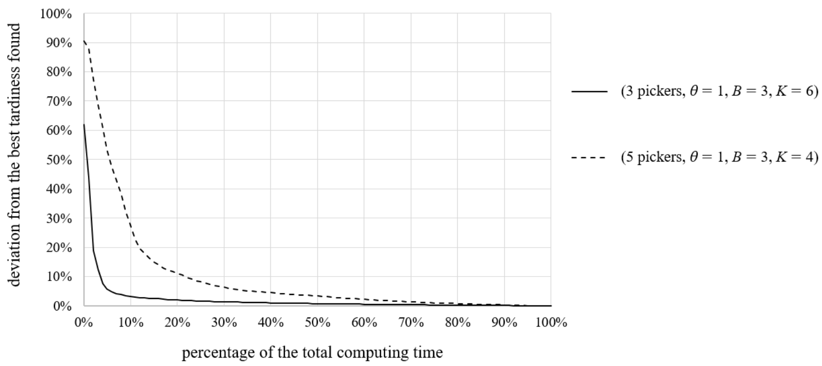


Fig. 6: Development of the solution quality over four hours of computing time

If three additional hours are spent on solving an instance from the class with 3 pickers, the total tardiness can be reduced by 1.6% on average. Thus, we conclude that the time limit of one hour is appropriate for those instances as the solution quality does not improve significantly. This is not true for problem instances from the class containing 5 order pickers. On average, the objective function value of the best solution found after one hour of computing time can be improved by 8.2%, which shows that the total tardiness can significantly be reduced by spending a larger amount of computing time. However, in the last hour, the reduction obtained amounts to less than 1.0%, indicating that four hours of computing time are sufficient in order to tackle problems with 5 order pickers. Summing up, it can be pointed out that the amount of computing time required for obtaining solutions of good quality increases with an increasing number of order pickers, which could be expected as the problem gets more complex when more pickers are available.

6 Conclusions and outlook

In this paper, we investigated the order assignment and sequencing, and vehicle routing problem (OASVRP), which is particularly pivotal for an efficient organization of the distribution processes in the retail industry. In the considered scenario, the orders are first processed in the warehouse by retrieving the respective requested items from their storage locations. After having completed the picking operations, vehicles will perform tours in order to deliver the requested items to the customers.

Order picking and vehicle routing operations are closely interconnected since a vehicle tour cannot start before all requested items of the orders included in the tour have been provided by the warehouse. Nevertheless, the integrated solution of these two subproblems has not been addressed in the literature so far. For solving the OASVRP, an algorithm of Ullrich (2013) could be adapted to this problem. However, the computational performance of this approach is limited. In order to introduce a more competitive approach, in particular for large problem instances, an iterated local search (ILS) algorithm for the OASRP has been proposed in this paper. Due to the complexity and the characteristics of the OASVRP, the improvement procedure includes two alternating variable neighborhood descent algorithms which tackle one subproblem each. By means of the ILS approach, the benefits from dealing with the OASVRP as a holistic problem could be investigated even for problem instances of a size encountered in practice.

Extensive numerical experiments have been conducted. In the first part of the experiments, it is demonstrated that the proposed ILS approach is suitable for solving large-sized instances. The second part of the experiments is devoted to the investigation of the benefit from integrating order picking and vehicle routing operations. It has been shown that the solution of the OASVRP as a holistic problem reduces the total tardiness by up to 88.7%, while the average reduction over all problem classes amounts to 37.8% compared to a sequential solution of the respective subproblems. Furthermore, problem characteristics have been pointed out under which a separate solution of the subproblems leads to acceptable results, and problem classes have been identified, where the consideration of the OASVRP as a holistic problem is inevitable in order to provide high-quality solutions.

Further research could concentrate on an extension of the problem regarding the picking operations. In this paper, the processing times of the orders can considered as given due to the assumption that customer orders have to be processed separately. However, the picking device may enable the order pickers to temporarily store a larger number of items, which allows for processing several customer orders on the same tour. In this case, it has to be decided which customer orders are included in a picker tour. Furthermore, the routes could not be determined in advance anymore, which makes the resulting problem even more complex. Nevertheless, integrating this aspect would clearly represent a worthwhile endeavor because the processing times in the warehouse would significantly decrease if customer orders do not have to be processed separately.

Regarding the vehicle routing subproblem, a straightforward extension represents the introduction of time windows for the delivery of the customers. In particular, early deliveries may cause problems when no room is available for temporarily storing the items. Such problems would be avoided by the integration of an earliest possible delivery date. A further interesting extension can be found in the consideration of vehicles with multiple compartments. Supermarkets receive several kinds of food, which have to be transported under different cooling conditions. These products can be transported on the same tour if the vehicle can be divided into different compartments, where each compartment represents a certain temperature zone (Hübner & Ostermeier, 2016).

References

Alidaee, B. & Rosa, D. (1997): Scheduling parallel machines to minimize total weighted and unweighted tardiness. *Computers & Operations Research* 24, 775–788.

Azi, N; Gendreau, M. & Potvin, J.-Y. (2007): An exact algorithm for a single-vehicle routing problem with time windows and multiple routes. *European Journal of Operational Research* 78, 755–766.

Azi, N; Gendreau, M. & Potvin, J.-Y. (2010): An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles. *European Journal of Operational Research* 202, 756–763.

Azi, N; Gendreau, M. & Potvin, J.-Y. (2014): An adaptive large neighborhood search for a vehicle routing problem with multiple routes. *Computers & Operations Research* 41, 167–173.

Azizoglu, M. & Kirca, O. (1998): Tardiness minimization on parallel machines. *International Journal of Production Economics* 55, 163–168.

Baker, K.R. & Bertrand, J.W.M. (1981): An investigation of due-date assignment rules with constrained tightness. *Journal of Operations Management* 1, 109–120.

Baker, K.R. & Bertrand, J.W.M. (1982): A dynamic priority rule for scheduling against due-dates. *Journal of Operations Management* 3, 37–42.

Biskup, D.; Herrmann, J. & Gupta, J.N.D. (2008): Scheduling identical parallel machines to minimize total tardiness. *International Journal of Production Economics* 115, 134–142.

Brah, S.A. (1996): A comparative analysis of due date based job sequencing rules in a flow shop with multiple processors. *Production Planning & Controls* 7, 362–373.

Brandão, J. & Mercer, A. (1997): A tabu search algorithm for the multi-trip vehicle routing and scheduling problem. *European Journal of Operational Research* 100, 180–191.

Brucker, P.; Hurink, J. & Werner, F. (1996): Improving local search heuristics for some scheduling problems. Part I. *Discrete Applied Mathematics* 65, 97–122.

Brucker, P.; Hurink, J. & Werner, F. (1997): Improving local search heuristics for some scheduling problems. Part II. *Discrete Applied Mathematics* 72, 47–69.

Cattaruzza, D.; Absi, N. & Feillet, D. (2016): The multi-trip vehicle routing problem with time windows and release dates. *Transportation Science* 50, 676–693.

Chen, Z.-L. & Vairaktarakis, G.L. (2005): Integrated scheduling of production and distribution operations. *Management Science* 51, 614–628.

Chen, Z.-L. (2010): Integrated production and outbound distribution scheduling: Review and extension. *Operations Research* 58, 130–148.

Chiang, W.-C. & Russell, R.A. (2004): A Metaheuristic for the Vehicle-Routeing Problem with Soft Time Windows. *The Journal of the Operational Research Society* 55, 1298–1310.

Congram, R.K.; Potts, C.N. & van de Velde, S.L. (2002): An iterated dynasearch algorithm for the single-machine total weighted tardiness scheduling problem. *INFORMS Journal on Computing* 14, 52–67.

Desaulniers, G.; Madsen, O.B.G. & Ropke, S. (2014):: The vehicle routing problem with time windows. *Vehicle routing: Problems, methods and applications* 2nd edition, Toth, P. & Vigo, D. (eds). Philadelphia: Society for Industrial and Applied Mathematics and the Mathematical Optimization Society, 119–160.

Dueck, G. & Scheuer, T. (1990): Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics* 1, 161–175.

DVZ – Deutsche Verkehrszeitung (2013): Wir liefern künftig im 24-Stunden-Rhythmus. URL: http://www.dvz.de/rubriken/logistik-verlader/single-view/nachricht/24-stunden-belieferung-ueber-alle-sortimente.html.

EDEKA Minden-Hannover: Zahlen, Daten & Fakten. URL: http://www.edeka-verbund.de/Unternehmen/de/edeka_minden_hannover/unternehmen_minden_hannover/zahlen_daten_fakten/zahlen_daten_fakten_minden_hannover.jsp

Fleischmann, B. (1990): The vehicle routing problem with multiple use of vehicles. Working paper, Fachbereich Wirtschaftswissenschaften, Universität Hamburg.

Fu, Z.; Eglese, R. & Li, L.Y.O. (2008): A unified tabu search algorithm for vehicle routing problems with soft time windows. *Journal of the Operational Research Society* 59, 663–673.

Gao, S.; Qi, L. & Lei, L. (2015): Integrated batch production and distribution scheduling with limited vehicle capacity. *International Journal of Production Economics* 160, 13–25.

Grosso, A.; Della Croce, F. & Tadei, R. (2004): An enhanced dynasearch neighborhood for the single-machine total weighted tardiness scheduling problem. *Operations Research Letters* 32, 68–72.

Gupta, J.N.D. & Maykut, A. R. (1973): Concepts, theory, and techniques –Scheduling jobs on parallel processors with dynamic programming. *Decision Sciences* 4, 447–457.

Hansen, P. & Mladenovic, N. (2001): Variable neighborhood search: principles and applications. *European Journal of Operational Research* 130, 449–467.

Henke, T; Speranza, M.G. & Wäscher, G. (2015): The multi-compartment vehicle routing problem with flexible compartment sizes. *European Journal of Operational Research* 246, 730–743.

Henn, S. & Schmid, V. (2013): Metaheuristics for order batching and sequencing in manual order picking systems. *Computers & Industrial Engineering* 66, 338–351.

Henn, S. (2015): Order batching and sequencing for the minimization of the total tardiness in picker-to-part warehouses. *Flexible Services and Manufacturing* 27, 86–114.

Ho, J.C. & Chang, Y.-L. (1991): Heuristic for minimizing mean tardiness of m parallel machines. *Naval Research Logistics* 38, 367–381.

Hurter, A.P. & Van Buer, M.G. (1996): The newspaper production/distribution problem. *Journal of Business Logistics* 17, 85–107.

Hübner, A. & Ostermeier, M. (2016): A multi-compartment vehicle routing problem with loading and unloading costs. Working Paper, Catholic University Eichstätt-Ingolstadt.

Koulamas, C. (1997): Decomposition and hybrid simulated annealing heuristics for the parallel-machine total tardiness problem. *Naval Research Logistics* 44, 109–125.

Li, C.-L.; Vairaktarakis, G.L. & Lee, C.-Y. (2005): Machine scheduling with deliveries to multiple customer locations. *European Journal of Operational Research* 164, 39–51.

Li, K.; Zhou, C.; Leung, J.Y.-T. & Ma, Y. (2016): Integrated production and delivery with single machine and multiple vehicles. *Expert Systems With Applications* 57, 12–20.

Liberatore, F.; Righini, G. & Salani, M. (2011): A column generation algorithm for the vehicle routing problem with soft time windows. *4OR* 9, 49–82.

Lourenço, H.R.; Martin, O.C. & Stützle, T. (2010): Iterated local search: Framework and applications. *Handbook of Metaheuristics*, 2nd edition., Gendreau, M. & Potvin, J.-Y. (eds.). International Series in Operations Research & Management Science 146. New York et al.: Springer, 363–397.

Low, C.; Li, R.-K. & Chang, C.-M. (2013): Integrated scheduling of production and delivery with time windows. *International Journal of Production Research* 51, 897–909.

Low, C.; Chang, C.-M.; Li, R.-K. & Huang, C.-L. (2014): Coordination of production scheduling and delivery problems with heterogeneous fleet. *International Journal of Production Research* 153, 139–148.

Olivera, A. & Viera, O. (2007): Adaptive memory programming for the vehicle routing problem with multiple trips. *Computers & Operations Research* 34, 28–47.

Petch, R & Salhi, S. (2004): A multi-phase constructive heuristic for the vehicle routing problem with multiple trips. *Discrete Applied Mathematics* 133, 69–92.

Pinedo, M.L. (2016): Scheduling: Theory, Algorithms, and Systems. 5th edition, Springer, Cham et al.

Polacek, M; Hartl, R.F. & Doerner, K. (2004): A variable neighborhood search for the multi depot vehicle routing problem with time windows. *Journal of Heuristics* 10, 613–627.

Prescott-Gagnon, E.; Desaulniers, G.; Drexl, M. & Rousseau, L.-M. (2010): European driver rules in vehicle routing with time windows. *Transportation Science* 44, 455–473.

Roodbergen, K. J. (2001): Layout and Routing Methods for Warehouses. Trial, Rotterdam.

Schmid, V.; Doerner, K.F. & Laporte, G. (2013): Rich routing problems arising in supply chain management. *European Journal of Operational Research* 224, 435–448.

Scholz, A.; Schubert, D. & Wäscher, G. (2016): Order picking with multiple pickers and due dates – Simultaneous solution of order batching, batch assignment and sequencing, and picker routing problems. Working Paper No. 5/2016, Faculty of Economics and Management, Otto-von-Guericke-Universität Magdeburg.

Shim, S.-O. & Kim, Y.-D. (2007): Scheduling on parallel identical machines to minimize total tardiness. *European Journal of Operational Research* 177, 135–146.

Taillard, É.D.; Laporte, G. & Gendreau, M. (1996): Vehicle routeing with multiple use of vehicles. *The Journal of the Operational Research Society* 47, 1065–1070.

Taillard, É.D.; Badeau, P.; Gendreau, M.; Guertin, F. & Potvin, J.-Y. (1997): A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportations Science* 31, 170–186.

Tarantilis, C.D.; Kiranoudis, C.T. & Vassiliadis, V.S. (2004): A threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem. *European Journal of Operational Research* 152, 148–158.

Ullrich, C.A. (2013): Integrated machine scheduling and vehicle routing with time windows. *European Journal of Operational Research* 227, 152–165.

Van Buer, M.G.; Woodruff, D.L. & Olson, R.T. (1999): Solving the medium newspaper production/distribution problem. *European Journal of Operational Research* 115, 237–253.

Vidal, T.; Crainic, T.G.; Gendreau, M. & Prins, C. (2013): Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *European Journal of Operational Research* 231, 1–21.

Yalaoui, F. & Chu, C. (2002): Parallel machine scheduling to minimize total tardiness. *International Journal of Production Economics* 76, 265–279.

Part IX:
Outlook on Further Research

Outlook on Further Research

Concerning the routing of order pickers, various promising areas for further research can be identified. In Part III and Part IV of this thesis, it has been shown that the application of a problem-specific formulation to the Picker Routing Problem in standard-aisle warehouses leads to a well-performing solution approach which can also easily be adapted to several constraints arising in practical applications. For example, the formulations can be used to construct simple tours, e.g. tours where each subaisle is visited at most once (Roodbergen, 2001). Simple tours are more straightforward and easier to memorize for the order pickers, reducing the risk of pick errors or deviations from the path (de Koster et al., 1999). However, simple tours are of course longer than optimal ones. By means of the model formulations, the benefit of using optimal tours instead of tours with a simple structure could be evaluated and problem settings could be identified where the generation of optimal tours is inevitable for keeping the tour lengths at a reasonable level. A further modification from practice can be found in the consideration of multiple deposit locations (de Koster & van der Poort, 1998). It would be interesting to investigate the impact of multiple deposit locations on the tour lengths and analyze whether the savings regarding the travel time justify the additional consolidation effort caused by the usage of multiple deposit locations.

For the Picker Routing Problem in narrow-aisle warehouses, a truncated branch-and-bound algorithm has been presented in Part V. This approach is based on two main assumptions which could be addressed in future research. First, order pickers follow given paths through the warehouse and are not permitted to deviate from their paths even though blocking situations may be avoided. By allowing pickers to deviate from their paths or even visit the pick locations in another sequence, the processing times of the orders may be reduced significantly. The second assumption is even more critical as deterministic travel velocities and pick times are dealt with. Since the tasks are performed by human operators, these components are not deterministic in practice (Parikh & Meller, 2010), and blocking situations cannot be identified in advance. Thus, it would be very important to take stochastic travel velocities and pick times into account. Arising blocking situations have to be predicted and robust tours have to be constructed where small fluctuations regarding these components do not cause additional blocking situations. Then, it would also be quite interesting to analyze the impact of the employment of an additional order picker on the average processing time of the orders. Furthermore, it could be evaluated whether it is really a good choice to use a narrow-aisle layout in order to maximize space utilization or if at least some standard aisles should be introduced.

Parts VI and VII deal with the Joint Order Batching and Picker Routing Problem for distance-related and tardiness-related objectives, respectively. The first-mentioned variant has been widely studied in the literature. However, tardiness-related objectives have been rarely considered when dealing with the Joint Order Batching and Picker Routing Problem, leaving several fields for further research. In this thesis, a variable neighborhood descent algorithm has been proposed in order to tackle this problem. The numerical experiments indicate that computing time is a very critical issue. Thus, further research could concentrate on the development of a fast approach which results in solutions of similar quality. Future research could also deal with the integration of further problem aspects. For example, priority values (weights) could be assigned to the customer orders and the weighted tardiness could be minimized. Furthermore, the on-line variant of the Joint Order Batching and Picker Routing Problem could be considered, i.e. customer orders are not available at the beginning but arrive over the planning horizon.

In Part VIII, a first solution approach to integrated order picking and vehicle routing is proposed. Since almost no research has been conducted in this field, basic aspects of the problem have been considered only. For example, it is assumed that order pickers are prohibited from processing more than one order on a single tour. Neglecting this assumption, the Order Batching Problem could be integrated here, resulting in much shorter average processing times and making due dates less difficult to be met. Regarding the vehicle routing part, the problem could be extended by taking a heterogeneous vehicle fleet or vehicles with different compartments into account (Henke et al., 2015; Hübner & Ostermeier, 2016). Furthermore, time windows with a hard lower bound could be introduced, i.e. vehicles cannot serve a customer before a certain point in time (Ullrich, 2013).

All problems dealt with in this thesis are based on the assumption that each article is assigned to exactly one storage location in the warehouse. This represents a very common assumption in the literature. However, in particular, when dealing with the Picker Routing Problem in narrow-aisle warehouses or with the Joint Order Batching and Picker Routing Problem, significant benefits can be expected from the assignment of frequently requested articles to multiple storage locations. In case of narrow-aisle warehouses, blocking situations can be avoided by having the possibility to choose between different locations. With respect to the Joint Order Batching and Picker Routing Problem, the consideration of multiple storage locations per article leads to a larger number of combinations of customer orders which result in promising picking orders. Thus, further research could concentrate on the design of exact and heuristic approaches to those problems taking multiple article locations into account, allowing for the evaluation of the benefit of assigning some articles to more than one storage location.

References

de Koster, R.; Roodbergen, K. J. & van Voorden, R. (1999): Reduction of Walking Time in the Center of de Bijenkorf. *New Trends in Distribution Logistics*, Speranza, M. G. & Stähly P., (eds.), 215-234, Springer: Berlin.

de Koster, R. & van der Poort, E. (1998): Routing Orderpickers in a Warehouse: A Comparison between Optimal and Heuristic Solutions. *IIE Transactions* 30, 469-480.

Henke, T; Speranza, M.G. & Wäscher, G. (2015): The Multi-Compartment Vehicle Routing Problem with Flexible Compartment Sizes. *European Journal of Operational Research* 246, 730-743.

Hübner, A. & Ostermeier, M. (2016): A Multi-Compartment Vehicle Routing Problem with Loading and Unloading Costs. Working Paper, Catholic University Eichstätt-Ingolstadt.

Parikh, P. J. & Meller, R. D. (2010): A Note on Worker Blocking in Narrow-Aisle Order Picking Systems when Pick Time is Non-Deterministic. *IIE Transactions* 42, 392-404.

Roodbergen, K. J. (2001): Layout and Routing Methods for Warehouses. Trial: Rotterdam.

Ullrich, C.A. (2013): Integrated Machine Scheduling and Vehicle Routing with Time Windows. *European Journal of Operational Research* 227, 152-165.