Das Pickup and Delivery Problem mit mehrdimensionalen Ladebeschränkungen

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.-Math. Dirk Männel
Geburtsdatum und -ort:	13.02.1972, Freiberg
Arbeit eingereicht am:	17.11.2017

Gutachter der schriftlichen Promotionsleistung:
apl. Prof. Dr. Andreas Bortfeldt
Prof. Dr. Lars Mönch

Datum der Disputation:	08.05.2018

Inhaltsverzeichnis

Teil I:

Zusammenfassung der Dissertation

Zusammenfassung der Dissertation

Die Untersuchung von Tourenplanungsproblemen ist von großem Interesse für Unternehmen der Logistikbranche, da diese an einer optimalen Auslastung ihrer Fahrzeuge und an der Realisierung möglichst kurzer Fahrtstrecken bzw. geringer Fahrtkosten interessiert sind. Aus diesem Grunde sind Probleme wie das „Capacitated Vehicle Routing Problem“ (CVRP) und das „Pickup and Delivery Problem“ (PDP) in der Vergangenheit in der Literatur detailliert untersucht worden (Toth & Vigo, 2014). Die klassische Modellierung dieser Tourenplanungsprobleme berücksichtigt dabei jedoch keine Nebenbedingungen hinsichtlich der Anordnung der geladenen Güter in den Laderäumen der Fahrzeuge. Solche Nebenbedingungen sind insbesondere im Zusammenhang mit dem Transport von Stückgütern (z.B. Kisten, Paletten, Gebinden, Materialrollen oder Fässern) praxisrelevant. Stückgüter werden heute zumeist in Containern oder in den Laderäumen von Lastkraftwagen transportiert. Dabei muss sichergestellt werden, dass für jeden Punkt einer Tour ein zulässiger Ladeplan existiert, der angibt, wie die einzelnen Packstücke in dem umhüllenden Container bzw. Laderaum positioniert werden sollen. Aus diesem Grund wurde in den letzten zehn Jahren ein Schwerpunkt der Forschung auf integrierte Tourenplanungs- und Packprobleme gelegt, bei denen die zu transportierenden Güter als zweidimensionale bzw. dreidimensionale Packstücke modelliert werden (Iori & Martello, 2013; Pollaris et al., 2015). Bei einer 3D-Modellierung können weitere Nebenbedingungen, die z.B. die Stabilität der Ladung oder die Zerbrechlichkeit einzelner Packstücke berücksichtigen, in die Problemformulierung einbezogen werden (Gendreau et al., 2006). Eine 2D-Modellierung ist hingegen in Fällen ausreichend, in denen eine Überstapelung von Packstücken aufgrund ihres hohen Gewichts oder ihrer Zerbrechlichkeit nicht möglich ist (Dominguez et al., 2016). Zusätzlich werden in vielen praktischen Anwendungsfällen LIFO-Bedingungen (Last In First Out) für Be- und Entladung berücksichtigt, d.h. zu entladende Packstücke müssen vom Fahrzeugheck aus frei zugänglich sein bzw. zu beladende Packstücke müssen an frei zugänglichen Positionen im Laderaum platziert werden (Bortfeldt et al., 2015). Gründe für die Berücksichtigung der LIFO-Bedingungen können z.B. enge Zeitpläne, fehlende Manpower, fehlende Ausrüstung oder der Transport von besonders schweren bzw. gefährlichen Gütern sein.

Im klassischen PDP besteht jeder Auftrag aus einem Ladegut mit gegebenem Gewicht welches von einem gewissen Beladeort zu einem gewissen Entladeort transportiert werden muss. Eine gegebene Anzahl von homogenen Fahrzeugen ist an einem einzigen Depot stationiert, d.h. alle Touren starten und enden dort. Das Depot sowie die Be- und Entladeorte bilden einen vollständigen Graphen mit gegebenen Distanzen. Die Touren müssen so gebildet werden, dass (i) jeder Be- und Entladeort genau einmal besucht wird, (ii) der Be- und Entladeort zu einem Auftrag in derselben Tour liegen und dies in der korrekten Reihenfolge, (iii) das Ladegewicht der Fahrzeuge zu keinem Zeitpunkt überschritten wird, (iv) eine gewisse Tourlänge bzw. Tourdauer nicht überschritten wird und (v) die Anzahl an Touren die gegebene Fahrzeuganzahl nicht übersteigt. Beim PDP wird zumeist die Minimierung der Tourenanzahl als primäres Zielkriterium

und die Minimierung der gesamten Fahrtkosten als sekundäres Zielkriterium verwendet (Li & Lim, 2003). Das PDP ist in der Vergangenheit ausführlich untersucht worden (Berbeglia et al., 2007; Parragh et al., 2008). Darüber hinaus berücksichtigen einige Autoren zusätzliche Nebenbedingungen wie Zeitfenster und Servicezeiten, inhomogene Fahrzeugflotten, Inkompatibilitäten bei den Ladeaufträgen und spezielle Arbeitszeitregeln für Fahrer (Xu et al., 2003). Im Zusammenhang mit Personentransporten wird das PDP als „Dial a Ride Problem“ (DARP) bezeichnet, dabei werden zusätzliche Nebenbedingungen für die Einhaltung der Servicequalität (z.B. maximal zulässige Reisedauer) in die Problemformulierung aufgenommen. Da das PDP ein NP-hartes Problem ist, kommen zur Lösung im Wesentlichen Metaheuristiken wie Simulated Annealing und Large Neighborhood Search (Bent & van Hentenryck, 2006), Adaptive Large Neighborhood Search (Ropke & Pisinger, 2006) und Guided Ejection Search (Nagata & Kobayashi, 2010) zum Einsatz. Exakte Lösungsansätze wurden von Ropke et al. (2007), Ropke & Cordeau (2009) sowie Baldacci et al. (2011) vorgestellt, dennoch stellen Metaheuristiken für größere PDP-Instanzen bis heute das Mittel der Wahl dar. Romero et al. (2007) erreichen in einem praktischen Anwendungsfall eine Kosteneinsparung von 10% durch den Einsatz eines genetischen Algorithmus.

Gendreau et al. (2006) formulierten erstmals das „Capacitated Vehicle Routing Problem with Three-Dimensional Loading Constraints“ (3L-CVRP), welches das klassische CVRP mit einem Containerladeproblem für dreidimensionale, quaderförmige Packstücke kombiniert und stellten einen aus zwei Tabu-Search-Algorithmen bestehenden Lösungsansatz vor. Bis heute wurden zahlreiche weitere Lösungsansätze für das 3L-CVRP veröffentlicht, u.a. von Fuellerer et al. (2010), Bortfeldt (2012), Ruan et al. (2013), Wei et al. (2014) und Zhang et al. (2015). Im Rahmen dieser Arbeit wird das PDP, welches eine Verallgemeinerung des CVRP darstellt, auf ähnliche Art und Weise mit dem Containerladeproblem kombiniert. Dabei wird bei der Modellierung des „Pickup and Delivery Problem with Three-Dimensional Loading Constraints“ (3L-PDP) darauf geachtet, dass sämtlicher Umladeaufwand, d.h. sämtliche Veränderungen von Packstückpositionen in den Laderäumen nach Verlassen des entsprechenden Beladeortes und vor Eintreffen am entsprechenden Entladeort, ausgeschlossen wird. Dafür ist die Einführung einer neuartigen Nebenbedingung, die im Folgenden „Umladeverbot“ genannt wird, notwendig.

Beim 3L-PDP bestehen die zu transportierenden Güter aus quaderförmigen Packstücken mit gegebener Länge, Breite und Höhe. Die Fahrzeuge besitzen einen quaderförmigen Laderaum, dessen Abmessungen ebenfalls gegeben sind. Für jeden Be- und Entladeort einer Tour muss ein zulässiger Packplan zur Verfügung gestellt werden, der Positionsangaben für alle an diesem Ort im Laderaum befindlichen Packstücke enthält. Dabei dürfen keine Packstücke aus dem Laderaum herausragen bzw. sich keine zwei Packstücke gegenseitig überlappen. Zusätzlich müssen die Packpläne die Bedingungen erfüllen, dass (i) jedes Packstück mit seinen Seitenflächen parallel zu den Seitenflächen des Laderaums platziert wird, (ii) die Höhenachse jedes Packstücks parallel zur Höhenachse des Laderaums liegt, (iii) ein gewisser Mindest-Prozentsatz der Grundfläche jedes Packstücks auf anderen Packstücken oder dem Boden des Laderaums aufliegt, (iv) auf zerbrechlichen Packstücken keine nicht zerbrechlichen Packstücke aufliegen, (v) die LIFO-Bedingungen an Beladeorten bzw. Entladeorten sowie (vi) das Umladeverbot eingehalten werden. Im Falle einer zweidimensionalen Modellierung (2L-PDP) sind entsprechend nur die Nebenbedingungen (i), (v) und (vi) zu beachten. Das 3L-PDP wurde bisher nur von Bartók & Imre (2011) untersucht, allerdings verzichten die Autoren auf eine Berücksichtigung der LIFO-Bedingungen und des Umladeverbots. Aufgrund der praktischen Relevanz dieser Nebenbedingungen ist ihre Berücksichtigung jedoch zwingend erforderlich. Im Rahmen der Analyse der numerischen Resultate des 3L-PDP zeigt sich, dass diese Nebenbedingungen großen Einfluss auf die Lösungsqualität haben. Zum 2L-PDP existiert eine Veröffentlichung von Malapert et al. (2008). Die Autoren schlagen hierbei einen Constraint Programming-Ansatz zur Erstellung der Packpläne vor, veröffentlichen jedoch keine numerischen Resultate.

In Männel & Bortfeldt (2016) werden verschiedene Varianten des 3L-PDP vorgestellt, die sich dahingehend unterscheiden, ob die LIFO-Bedingungen und das Umladeverbot berücksichtigt werden oder nicht. Zusätzlich werden zwei grundlegende Konzepte vorgestellt, wie die Berücksichtigung von LIFO-Bedingungen und Umladeverbot erfolgen kann, nämlich (i) durch eine Beschränkung auf Touren eines gewissen Layouts („Independent Partial Routes“-Ansatz, kurz IPR) oder (ii) durch eine neuartige Packprozedur, die Packpläne für mehrere Punkte einer Tour gleichzeitig erstellen kann („Interrelated Packing“-Ansatz, kurz IP). In Männel & Bortfeldt (2016) wird der IPR-Ansatz im Detail untersucht, während der IP-Ansatz dem Beitrag von Männel & Bortfeldt (2018) vorbehalten bleibt.

Da das 2L-PDP wie das 3L-PDP aus der Kombination zweier NP-schwerer Probleme besteht, die in der Praxis zumeist mit Metaheuristiken gelöst werden, wird zu ihrer Lösung ein Hybridansatz gewählt, bei dem eine „äußere“ Prozedur zur Tourenplanung mit einer „inneren“ Prozedur für die Packprüfung kombiniert wird. Dieser Hybridansatz kann auch zur Lösung weiterer Tourenplanungsprobleme mit mehrdimensionalen Ladebeschränkungen eingesetzt werden (Bortfeldt et al., 2015). In Männel & Bortfeldt (2016) wird eine Ausgestaltung des Hybrid-Algorithmus zur Lösung des 3L-PDP vorgestellt, die ein Large Neighborhood Search-Verfahren (LNS) zur Tourenplanung benutzt und einen Tree Search Algorithmus (TRS) zur Erstellung der Packpläne verwendet. Beide Teilprozeduren sind bewährte Verfahren. Die Tourenplanungsprozedur basiert im Wesentlichen auf dem Lösungsansatz von Ropke & Pisinger (2006) für das klassische PDP. Die LNS-Prozedur benutzt vier Remove-Operatoren und drei Insert-Operatoren (Heuristiken), die abwechselnd mit festen Auswahlwahrscheinlichkeiten eingesetzt werden. Dabei entfernt der Remove-Operator einen gewissen Teil der Aufträge aus ihren Touren, während der Insert-Operator diese Aufträge wieder in andere Touren bzw. an anderen Positionen innerhalb derselben Tour einfügt, um so eine Reduzierung der Fahrtstrecke zu erreichen. Als Akzeptanzkriterium wird das Kriterium des Simulated Annealing mit einem geometrischen Kühlschema verwendet. Die Anzahl der zu entfernenden Aufträge wird bei jeder Iteration stochastisch gewählt, um einerseits eine gute Diversifikation zu erreichen und andererseits die Suche in der „Nähe“ der aktuell besten Lösung intensivieren zu können. Für den Fall, dass die

Startlösung unzulässig ist, weil nicht alle Aufträge mit der zur Verfügung stehenden Anzahl an Fahrzeugen bedient werden können, werden temporär unzulässige Lösungen zugelassen. Bis die Zulässigkeit hergestellt ist, wird ein passender Strafterm in die Zielfunktion aufgenommen. Die TRS-Packprozedur wurde bereits in Bortfeldt (2012) zur Lösung des 3L-CVRP eingesetzt und hat sich als sehr leistungsfähiges Packverfahren im 3D-Fall erwiesen. Dabei handelt es sich um eine unvollständige Baumsuche, die nach dem Prinzip der Tiefensuche implementiert ist und durch eine rekursive Prozedur ausgeführt wird. Jeder Knoten im Suchbaum wird dabei im Wesentlichen durch drei Elemente charakterisiert: (i) einen unvollständigen Packplan, der bereits Platzierungen für einige Packstücke enthält, (ii) die Menge noch nicht platzierter Packstücke und (iii) die Kandidatenliste möglicher weiterer Platzierungen. Wenn für ein noch nicht platziertes Packstück keine mögliche Platzierung mehr vorhanden ist, wird die Suche im aktuellen Knoten des Baums abgebrochen und zum übergeordneten Knoten zurückgekehrt, da dann in den meisten Fällen keine vollständige Lösung mehr erreicht werden kann. Der Suchalgorithmus wird beendet, wenn ein vollständiger Packplan erzeugt wurde oder wenn eine gewisse Obergrenze an Aufrufen der rekursiven Prozedur ohne Erfolg überschritten wurde. Zusätzlich werden spezielle Maßnahmen zur Verbesserung der Performance des Algorithmus getroffen, z.B. wird ein Cache verwendet, in dem alle bereits geprüften Packstückfolgen abgelegt werden. In Männel & Bortfeldt (2016) wird im Tourenplanungsmodul eine Einschränkung auf sogenannte IPR-Touren vorgenommen. IPR-Touren müssen dabei die Bedingung erfüllen, dass (i) Entladeorte in der Tour in umgekehrter Reihenfolge zu ihren korrespondierenden Beladeorten liegen und (ii) ein Beladeort nur angefahren werden darf, wenn das Fahrzeug zu diesem Zeitpunkt leer ist oder der vorhergehende Ort in der Tour ebenfalls ein Beladeort ist. Es konnte gezeigt werden, dass durch die Beschränkung auf IPR-Touren die LIFO-Bedingung an Entladeorten und das Umladeverbot automatisch erfüllt sind, währenddessen die LIFO-Bedingung an Beladeorten von der TRS-Prozedur sichergestellt wird. Des Weiteren wurde gezeigt, dass eine Packprüfung nur an sogenannten letzten Beladeorten (d.h. Beladeorten, auf die ein Entladeort folgt) durchgeführt werden muss und dass die Packprüfungen für die letzten Beladeorte einer IPR-Tour unabhängig voneinander mit der TRS-Packprozedur erfolgen können. Für alle weiteren Be- und Entladeorte einer Tour können die benötigten Packpläne aus den Packplänen der letzten Beladeorte abgeleitet werden. Für den Test des Hybrid-Algorithmus wurden 54 3L-PDP-Testinstanzen mit bis zu 100 Aufträgen und bis zu 300 Packstücken erstellt. Die Testergebnisse sind plausibel und zeigen den erwarteten Trade-off zwischen Fahrtstrecke und Umladeaufwand. Es wurde festgestellt, dass die Fahrtstrecke im Mittel über alle Testinstanzen um ca. 12% geringer ausfällt, wenn die LIFO-Bedingungen und das Umladeverbot nicht berücksichtigt werden. In diesem Fall muss die Einsparung an Fahrtstrecke aber mit einem Umladeaufwand, d.h. der Notwendigkeit, gewisse Packstücke während der Touren im Laderaum umzupositionieren, „bezahlt" werden.

Die IPR-Bedingung sorgt beim 3L-PDP für eine starke Beschränkung bei der Bildung der Touren und behindert somit das Auffinden von besonders guten Lösungen. In Männel & Bortfeldt (2018) liegt das Hauptaugenmerk daher auf dem „Interrelated Packing“-Ansatz, bei dem die IPR-Bedingung des vorangegangenen Lösungsansatzes entfällt. Im Fall des IP-Ansatzes müssen damit die Einhaltung der LIFO-Bedingungen und des Umladeverbots von der Packprozedur sichergestellt werden. Es hat sich gezeigt, dass dazu ein neuartiger Typ von Packprozedur nötig ist, der für mehrere letzte Beladeorte einer Tour gleichzeitig Packpläne erstellen kann um auf diese Weise das Umladeverbot sicherzustellen. Die TRS-Packprozedur wird zu diesem Zweck so erweitert, dass bei einer Packprüfung für einen gewissen letzten Beladeort einer Tour alle Platzierungen von Packstücken, die sich bereits an vorhergehenden letzten Beladeorten im Fahrzeug befanden, aus den Packplänen dieser Orte übernommen werden. Falls für den aktuell betrachteten letzten Beladeort nach einer gewissen Anzahl von Versuchen kein zulässiger Packplan gefunden werden konnte, geht der Backtracking-Mechanismus der rekursiven Prozedur dann zu einem vorhergehenden letzten Beladeort der Tour zurück und ändert Positionen für dort bereits eingeladene Packstücke. Anschließend müssen alle nachfolgenden letzten Beladeorte erneut geprüft werden. Die Packprüfung endet erfolgreich, wenn für alle letzten Beladeorte der Tour ein zulässiger Packplan gefunden wurde und somit das Umladeverbot sichergestellt ist. Bei den numerischen Tests des IP-Ansatzes konnte eine deutliche Verbesserung der Lösungsqualität gegenüber dem einfacheren IPR-Lösungsansatz festgestellt werden, während der Rechenaufwand durch die erheblich komplexere Packprozedur ebenfalls deutlich anstieg.

In Männel (2017)[1] wird der gewählte hybride Lösungsansatz für das 3L-PDP auf das 2L-PDP übertragen. Es wird dieselbe LNS-Tourenplanungsprozedur wie für das 3L-PDP benutzt. Da die TRS-Packprozedur im zweidimensionalen Fall jedoch deutlich einfacheren Heuristiken kaum überlegen ist, wird für das 2L-PDP eine einfachere Packprozedur verwendet, die sechs konstruktive Packheuristiken benutzt. Diese Heuristiken wurden bereits erfolgreich für das 2L-CVRP verwendet (Zachariadis et al. 2009; Leung et al. 2011), darunter sind weithin bekannte Packverfahren wie „Bottom-Left Fill“ (Chazelle, 1983) oder „Touching Perimeter“ (Lodi et al. 1999). Beim 2L-PDP besteht wie beim 3L-PDP die Problematik, dass die LIFO-Bedingungen für Be- und Entladeorte und das Umladeverbot sichergestellt werden müssen. Dies geschieht zunächst mit dem bereits vorgestellten IPR-Ansatz aus Männel & Bortfeldt (2016), der für das 2L-PDP unverändert übernommen werden kann. Dabei erfolgen voneinander unabhängige Packprüfungen für jeden letzten Beladeort einer Tour. Die Packprüfungen werden so ausgeführt, dass die sechs konstruktiven Packheuristiken mit fünf Packstückreihenfolgen kombiniert werden, wodurch für jeden letzten Beladeort maximal 30 Versuche unternommen werden, einen zulässigen Packplan zu erstellen. Um dem Hauptnachteil des IPR-Ansatzes, nämlich der starken Beschränkung bei der Tourenbildung zu begegnen, wird ein weiterer Lösungsansatz für das 2L-PDP präsentiert. Bei dem sogenannten „Simultaneous Packing“-Ansatz (kurz SP), werden in der Tourenplanungsprozedur ausschließlich sogenannte LIFO-Touren berücksichtigt, d.h. Touren bei denen die Entladeorte in jeweils umgekehrter Reihenfolge zu ihren korrespondierenden Beladeorten liegen. Die Bedingung für LIFO-Touren stellt damit eine wesentliche Abschwächung

[1] Zur Begutachtung bei Annals of Operations Research eingereicht

der IPR-Bedingung dar. Es konnte gezeigt werden, dass eine LIFO-Tour im Sinne des 2L-PDP zulässig ist, wenn (i) für jeden letzten Beladeort der LIFO-Tour ein zulässiger Packplan gefunden werden kann und (ii) für alle letzten Beladeorte der Tour der entsprechende Packplan mit ein und derselben Packheuristik sowie ein und derselben Packstückreihenfolge erzeugt werden kann. Grund hierfür ist, dass die konstruktiven Packheuristiken keine stochastischen Elemente enthalten und nicht „vorausschauend“ arbeiten. Das bedeutet, dass die Positionierung eines Packstückes im Laderaum nur von den Eigenschaften des Packstückes selbst und von denen früher eingeladener Packstücke abhängt, jedoch nicht von denen später eingeladener Packstücke. Somit ist das Umladeverbot unter den gegebenen Voraussetzungen immer erfüllt. Die Bedingung (ii) kann dabei noch so abgeschwächt werden, dass die gleiche Packheuristik und die gleiche Packstückreihenfolge nur für solche letzten Beladeorte verwendet werden müssen, bei denen der Fahrzeugladeraum zwischen diesen Beladeorten nicht mindestens einmal komplett geleert wird, d.h. bei denen sich mindestens ein gleiches Packstück im Laderaum befindet. Die Packprozedur beim „Simultaneous Packing“-Ansatz arbeitet folglich so, dass sie die Packprüfungen für alle letzten Beladeorte einer Tour, die besagte Bedingung erfüllen, simultan (gleichzeitig) ausführt. Die Packprozedur wird erst beendet, wenn es gelungen ist, mit einer einzigen Kombination von Packheuristik und Packstückreihenfolge zulässige Packpläne für alle relevanten letzten Beladeorte zu erzeugen oder wenn alle 30 Kombinationen aus Packheuristik und Packstückreihenfolge erfolglos probiert worden sind. Für die numerischen Tests des Hybrid-Algorithmus wurden 60 2L-PDP-Testinstanzen auf der Basis bekannter 2L-CVRP-Instanzen (Gendreau et al. 2008) erzeugt. Bei den Testergebnissen zeigt der SP-Ansatz wie erwartet eine deutlich bessere Lösungsqualität als der einfachere IPR-Lösungsansatz. Analog zum 3L-PDP zeigt sich, dass das Ausschließen allen Umladeaufwands durch Einhaltung der LIFO-Bedingungen und des Umladeverbots einen deutlichen Einfluss auf die Lösungsqualität hat. In der Zukunft sollte versucht werden, die Packprozedur des SP-Ansatzes so weiterzuentwickeln, dass mit ihr zusätzlich auch die LIFO-Bedingung an Entladeorten geprüft werden kann. Damit könnte die verbliebene Einschränkung bei der Tourenplanung (Beschränkung auf LIFO-Touren) eliminiert werden, um so eine noch bessere Lösungsqualität zu erreichen.

Literaturverzeichnis

Baldacci, R; Bartolini, E; Mingozzi, A (2011): An Exact Algorithm for the Pickup and Delivery Problem with Time Windows. *Operations Research*, 59(2):414–426.

Bartók, T; Imre, C (2011): Pickup and Delivery Vehicle Routing with Multidimensional Loading Constraints. *Acta Cybernetica*, 20(1):17–33.

Bent, R; van Hentenryck, P (2006): A Two-Stage Hybrid Algorithm for Pickup and Delivery Vehicle Routing Problems with Time Windows. *Computers & Operations Research*, 33(4):875–893.

Berbeglia, G; Cordeau, JF; Gribkovskaia, I; Laporte, G (2007): Static Pickup and Delivery Problems: A Classification Scheme and Survey. *Top*, 15(1):1–31.

Bortfeldt, A (2012): A Hybrid Algorithm for the Capacitated Vehicle Routing Problem with Three-Dimensional Loading Constraints. *Computers & Operations Research*, 39(9):2248–2257.

Bortfeldt, A; Hahn, T; Männel, D; Mönch, L (2015): Hybrid Algorithms for the Vehicle Routing Problem with Clustered Backhauls and 3D Loading Constraints. *European Journal of Operational Research*, 243(1):82–96.

Chazelle, B (1983): The Bottom-Left Bin-Packing Heuristic: An Efficient Implementation. *IEEE Transactions on Computers*, C-32(8):697–707.

Dominguez, O; Guimarans, D; Juan, AA; De La Nuez, I (2016): A Biased-Randomised Large Neighbourhood Search for the Two-Dimensional Vehicle Routing Problem with Backhauls. *European Journal of Operational Research*, 255(2):442–462.

Fuellerer, G; Doerner, KF; Hartl, RF; Iori, M (2010): Metaheuristics for Vehicle Routing Problems with Three-Dimensional Loading Constraints. *European Journal of Operational Research*, 201(3):751–759.

Gendreau, M; Iori, M; Laporte, G; Martello, S (2006): A Tabu Search Algorithm for a Routing and Container Loading Problem. *Transportation Science*, 40(3):342–350.

Gendreau, M; Iori, M; Laporte, G; Martello, S (2008): A Tabu Search Heuristic for the Vehicle Routing Problem with Two-Dimensional Loading Constraints. *Networks,* 51(1):4–18.

Iori, M; Martello, S (2013): An Annotated Bibliography of Combined Routing and Loading Problems. *Yugoslav Journal of Operations Research*, 23(3):311–326.

Leung, SCH; Zhou, X; Zhang, D; Zheng, J (2011): Extended Guided Tabu Search and a New Packing Algorithm for the Two-Dimensional Loading Vehicle Routing Problem. *Computers & Operations Research*, 38(1):205–215.

Li, H; Lim, A (2003): A Metaheuristic for the Pickup and Delivery Problem with Time Windows. *International Journal on Artificial Intelligence Tools*, 12(2):173–186.

Lodi, A; Martello, S; Vigo, D (1999): Heuristic and Metaheuristic Approaches for a Class of Two-Dimensional Bin Packing Problems. *INFORMS Journal on Computing*, 11(4):345–357.

Malapert, A; Guéret, C; Jussien, N; Langevin, A; Rousseau, LM (2008): Two-Dimensional Pickup and Delivery Routing Problem with Loading Constraints. *Proceedings of the First CPAIOR Workshop on Bin Packing and Placement Constraints (BPPC'08)*, Paris, France.

Männel, D (2017): Hybrid Algorithms for the Vehicle Routing Problem with Pickup and Delivery and Two-Dimensional Loading Constraints. Working Paper No. 10/2017, Faculty of Economics and Management, Otto-von-Guericke University Magdeburg.

Männel, D; Bortfeldt, A (2016): A Hybrid Algorithm for the Vehicle Routing Problem with Pickup and Delivery and Three-Dimensional Loading Constraints. *European Journal of Operational Research*, 254(3):840–858.

Männel, D; Bortfeldt, A (2018): Solving the Pickup and Delivery Problem with Three-Dimensional Loading Constraints and Reloading Ban. *European Journal of Operational Research*, 264(1):119–137.

Nagata, Y; Kobayashi, S (2010): A Memetic Algorithm for the Pickup and Delivery Problem with Time Windows Using Selective Route Exchange Crossover. *Parallel Problem Solving from Nature, PPSN XI*, R. Schaefer et al. (eds.), 536–545, Springer: Berlin.

Parragh, SN; Doerner, KF; Hartl, RF (2008): A Survey on Pickup and Delivery Problems. Part II: Transportation between Pickup and Delivery Locations. *Journal für Betriebswirtschaft*, 58(2):81–117.

Pollaris, H; Braekers, K; Caris, A; Janssens, G; Limbourg, S (2015): Vehicle Routing Problems with Loading Constraints: State-of-the-Art and Future Directions. *OR Spectrum*, 37(2):297–330.

Romero, M; Sheremetov, L; Soriano, A (2007): A Genetic Algorithm for the Pickup and Delivery Problem: An Application to the Helicopter Offshore Transportation. In: Castillo, O. (Ed.): Theoretical Advances and Applications of Fuzzy Logic, ASC 42, Springer, Berlin etc., 435–444.

Ropke, S; Cordeau, JF (2009): Branch and Cut and Price for the Pickup and Delivery Problem with Time Windows. *Transportation Science*, 43(3):267–286.

Ropke, S; Cordeau, JF; Laporte, G. (2007): Models and Branch-and-Cut Algorithms for Pickup and Delivery Problems with Time Windows. *Networks*, 49(4):258–272.

Ropke, S; Pisinger, D (2006): An Adaptive Large Neighborhood Search for the Pickup and Delivery Problem with Time Windows. *Transportation Science*, 40(4):455–472.

Ruan, Q; Zhang, Z; Miao, L; Shen, H (2013): A Hybrid Approach for the Vehicle Routing Problem with Three-Dimensional Loading Constraints. *Computers & Operations Research,* 40(6):1579–1589.

Toth, P; Vigo, D (2014): Vehicle Routing: Problems, Methods, and Applications, second edition. MOS-SIAM series on optimization, Philadelphia.

Wei, L; Zhang, Z; Lim, A (2014): An Adaptive Variable Neighborhood Search for a Heterogeneous Fleet Vehicle Routing Problem with Three-Dimensional Loading Constraints. *IEEE Computational Intelligence Magazine*, 9(4):18–30.

Xu, H; Chen, ZL; Rajagopal, S; Arunapuram, S (2003): Solving a Practical Pickup and Delivery Problem. *Transportation Science*, 37(3):347–364.

Zachariadis, EE; Tarantilis, CD, Kiranoudis, CT (2009): A Guided Tabu Search for the Vehicle Routing Problem with Two-Dimensional Loading Constraints. *European Journal of Operational Research*, 195(3):729–743.

Zhang, Z; Wei, L; Lim, A (2015): An Evolutionary Local Search for the Capacitated Vehicle Routing Problem Minimizing Fuel Consumption under Three-Dimensional Loading Constraints. *Transportation Research Part B*, 82:20–35.

Teil II:

A Hybrid Algorithm for the Vehicle Routing Problem with Pickup and Delivery and Three-Dimensional Loading Constraints

European Journal of Operational Research 254 (2016) 840–858

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Production, Manufacturing and Logistics

A hybrid algorithm for the vehicle routing problem with pickup and delivery and three-dimensional loading constraints

Dirk Männel, Andreas Bortfeldt[*]

Otto von Guericke University, Universitätsplatz 2, 39106 Magdeburg, Germany

ARTICLE INFO

Article history:
Received 6 July 2015
Accepted 7 April 2016
Available online 19 April 2016

Keywords:
Transportation
Vehicle routing
Packing
Pickup and delivery

ABSTRACT

In this paper, we extend the classical Pickup and Delivery Problem (PDP) to an integrated routing and three-dimensional loading problem, called PDP with three-dimensional loading constraints (3L-PDP). We are given a set of requests and a homogeneous fleet of vehicles. A set of routes of minimum total length has to be determined such that each request is transported from a loading site to the corresponding unloading site. In the 3L-PDP, each request is given as a set of 3D rectangular items (boxes) and the vehicle capacity is replaced by a 3D loading space. We investigate which constraints will ensure that no *re*loading effort will occur, i.e. that no box is moved after loading and before unloading. A spectrum of 3L-PDP variants is introduced with different characteristics in terms of reloading effort. We propose a hybrid algorithm for solving the 3L-PDP consisting of a routing and a packing procedure. The routing procedure modifies a well-known large neighborhood search for the 1D-PDP. A tree search heuristic is responsible for packing boxes. Computational experiments were carried out using 54 newly proposed 3L-PDP benchmark instances.

1. Introduction

Routing vehicles and loading them with goods represent two major challenges in transportation logistics. Routing and loading problems have to be tackled as integrated problems if companies are interested in optimizing both the routing of vehicles and the corresponding loading of goods. Gendreau, Iori, Laporte, and Martello (2006) first formulated and solved an integrated routing and loading problem, namely the capacitated vehicle routing problem (CVRP) with three-dimensional (3D) loading constraints (3L-CVRP). Contrasting to the classical CVRP, customer demands are represented as sets of parallel-epipeds (called boxes) and the scalar capacity of a vehicle is replaced by a 3D rectangular loading space. This essential modification allows for a more detailed modeling of mixed cargo transportation by vehicles. Several packing constraints, e.g. concerning stacking of goods, can only be considered if customer demands are viewed as sets of 3D items. To ensure that calculated routes can actually be implemented, a 3D modeling of cargo and loading spaces is in many situations indispensable (see Bortfeldt & Homberger, 2013). Thus it seems to be desirable to model and solve further types of vehicle routing problems (VRPs) as integrated routing and 3D loading problems (3L-VRP).

This task is tackled here for the classical Pickup and Delivery Problem (PDP). In the classical PDP, we are given a set of transportation requests that have to be served by a fleet of homogeneous vehicles with a uniform 1D capacity. Each request is characterized by a 1D demand that has to be transported from a *specific* loading site (pickup point) to a *specific* unloading site (delivery point). Since we have a single pickup point and single delivery point per request, the classical PDP belongs to the one-to-one VRPs with pickup and delivery. A set of routes, each starting and ending at the single depot, has to be constructed in such a way that (i) each request is served at only one route and its pickup point is visited before its delivery point; (ii) the capacity of a used vehicle is never exceeded by the set of loaded goods; (iii) the length of each route does not exceed a given limit; (iv) the number of routes does not exceed the given number of vehicles, and (v) the transportation cost, given by the total travel distance, is minimized.

To extend the classical PDP to an integrated routing and 3D loading problem, called hereafter PDP with three-dimensional loading constraints (3L-PDP), the demands are taken as sets of 3D rectangular items and the vehicles are equipped by a 3D rectangular loading space. As usual for the 3L-CVRP, we want to have a problem formulation that rules out any *re*loading effort. That is, the boxes should not be moved *after* loading and *before* unloading. In the 3L-CVRP this is guaranteed by the so-called Last-In-First-Out

[*] Corresponding author. Tel.: +49 391 6711842.
E-mail addresses: dirk.maennel@gmx.de (D. Männel), andreas.bortfeldt@fernuni-hagen.de, andreas.bortfeldt@ovgu.de (A. Bortfeldt).

http://dx.doi.org/10.1016/j.ejor.2016.04.016
0377-2217/© 2016 Elsevier B.V. All rights reserved.

Table 1
Sample of heuristics for the classical PDP.

Reference	Type of heuristic
Nanry and Barnes (2000)	Reactive tabu search
Li and Lim (2001)	Tabu embedded simulated annealing
Lim, Lim, and Rodrigues (2002)	Squeeky wheel optimization
Pankratz (2005)	Grouping genetic algorithm
Lu and Dessouky (2006)	Construction heuristic
Bent and van Hentenryck (2006)	Hybrid algorithm: simulated annealing, large neighborhood search
Ropke and Pisinger (2006)	Adaptive large neighborhood search
Derigs and Döhmer (2008)	Indirect local search with greedy decoding
Nagata and Kobayashi (2010)	Guided ejection search

(LIFO) condition. It turns out that this constraint is not sufficient to eliminate any reloading effort for the 3L-PDP. Therefore, additional constraints are introduced for this purpose. This leads to a spectrum of 3L-PDP variants that are afterwards defined more formally.

A hybrid algorithm for solving the 3L-PDP is proposed that is composed of the modified large neighborhood search (LNS) algorithm by Ropke and Pisinger (2006) for the 1D-PDP and the tree search (TRS) algorithm for packing boxes by Bortfeldt (2012). The hybrid algorithm is tested by means of 54 newly introduced 3L-PDP benchmark instances with up to 100 requests.

The rest of the paper is organized as follows: Section 2 reviews the relevant literature. In Section 3 crucial features of the 3L-PDP are discussed and some variants of the 3L-PDP are formulated. Section 4 describes the hybrid algorithm, while in Section 5 numerical results of experiments are presented and analyzed. Conclusions are drawn and an outlook at further research is given in Section 6.

2. Related work

In our literature review, we will focus on recent work on the classical PDP with paired pickup and delivery points and on VRPs with loading constraints. In particular, we will consider recent papers on PDP with loading constraints. Moreover, we look for practical applications of 3L-PDP dealt with in this paper. We refer the reader to Toth and Vigo (2014) for a comprehensive survey on vehicle routing.

2.1. Solution methods for the classical PDP

In pickup and delivery problems, goods or passengers are transported between customers or institutions. Following the classification schema by Parragh, Doerner, and Hartl (2008) the classical PDP is characterized by paired pickup and delivery points, i.e. each pickup point is generally associated with a special delivery point and vice versa. Moreover, the PDP deals with the transportation of goods; hence, no special constraints and objectives are involved concerning the (in)convenience of passengers as in dial-a-ride problems. A further distinction can be made with regard to the number of available vehicles and we will consider only the multi vehicle case, while the single vehicle case, representing an immediate extension of the Traveling Salesman Problem (TSP), is not considered here.

Mathematical models of the classical PDP or PDP with time windows (PDPTW) can be found, e.g. in Parragh et al. (2008) and in Toth and Vigo (2014). Most of the published solution methods are surveyed by Berbeglia, Cordeau, Gribkovskaia, and Laporte (2007) and Parragh et al. (2008). The PDP is NP-hard, as it generalizes the TSP. Therefore, mainly classical heuristics and metaheuristics were proposed for solving the PDP. A representative sample of recent heuristics is listed in Table 1. For further details of the algorithms, the reader is referred to the references; some comments can be found in Parragh et al. (2008) and in Toth and Vigo (2014). For an introduction in metaheuristic approaches we refer to Gendreau and Potvin (2010).

All solution methods listed in Table 1 are developed for the PDPTW, i.e. time windows are always considered. However, service times are only taken into account by Li and Lim (2001), Nanri and Barnes (2000) and Ropke and Pisinger (2006). Almost all methods minimize the routing cost (total travel distance) and several methods do also minimize the number of routes. The multi depot case is only handled by Ropke and Pisinger (2006). Almost all methods of Table 1 assume that the vehicle fleet is homogeneous. The case of heterogeneous vehicles is dealt with by Xu, Chen, Rajagopal, and Arunapuram (2003) and Ropke and Pisinger (2006). Most of the solution methods listed in Table 1 are evaluated by means of the benchmark instances proposed by Li and Lim (2001). Outstanding results especially for larger instances were achieved through the neighborhood search methods by Bent and van Hentenryck (2006) and Ropke and Pisinger (2006), while the method of Li and Lim (2001) proved to be very successful for smaller instances.

A branch and cut algorithm for the PDPTW was proposed by Ropke, Cordeau, and Laporte (2007), while Ropke and Cordeau (2009) described a branch and cut and price algorithm. Baldacci, Bartolini, and Mingozzi (2011) recently presented an exact algorithm based on a set-partitioning-like integer formulation. These exact PDPTW algorithms are capable of solving PDPTW instances with up to 500 requests; nevertheless, the numerical results reveal that heuristic approaches remain indispensable for large PDP instances.

2.2. Vehicle routing problems with loading constraints

Iori and Martello (2010, 2013) and Pollaris, Braekers, Caris, Janssens, and Limbourg (2015) survey the state of the art in the field of integrated vehicle routing and loading problems. Generally, the literature is still limited and this applies in particular to VRPs with 3D loading constraints (3L-VRP).

The 3L-CVRP was introduced by Gendreau et al. (2006) with five additional packing constraints frequently occurring in freight transportation. These include a last-in-first-out (LIFO) loading constraint, a weight constraint, an orientation constraint, a support constraint, and a stacking constraint (see Section 3 for details). Gendreau et al. suggest a two-stage tabu search algorithm for solving the 3L-CVRP. The “outer” tabu search serves for planning the routes, while the “inner” tabu search solves a 3D strip packing problem in order to load a vehicle according to a given customer sequence. Tarantilis, Zachariadis, and Kiranoudis (2009) propose a hybrid procedure combining the strategies tabu search and guided local search. They use a collection of plain packing heuristics. Fuellerer, Doerner, Hartl, and Iori (2010) develop an ant colony algorithm for routing that is integrated with fast but effective packing heuristics. Wang, Guo, Chen, Zhu, and Lim (2010) design a two-phase tabu search algorithm for routing that cooperates with two constructive packing heuristics (see also Zhu, Qin, Lim, & Wang, 2012). Wisniewski, Ritt, and Buriol (2011) propose a tabu search for routing and a randomized bottom left-based packing algorithm. Bortfeldt (2012) suggests a hybrid algorithm for the 3L-CVRP with a tabu search procedure for routing and a tree search algorithm for loading vehicles. Ruan, Zhang, Miao, and Shen (2013) present a honey bee mating algorithm for routing that is combined with six loading heuristics. Lacomme, Toussaint, and Duhamel (2013) propose an effective hybrid procedure for the 3L-CVRP that, however, does not consider all 3D packing constraints introduced by Gendreau et al. (2006). Tao and Wang (2015) developed a tabu search procedure and hybridized it with an effective packing algorithm. Very recently, Wei, Zhang, and Lim (2014) proposed an adaptive variable neighborhood search algorithm and Zhang, Wei, and Lim (2015) suggest an evolutionary local search method.

Moura and Oliveira (2009) introduce the VRP with time windows and 3D loading constraints (3L-VRPTW) with two objectives and present two heuristic procedures for this problem. The number of vehicles is minimized with higher priority, whereas the total travel distance is minimized with lower priority. The authors do not consider the weight and the stacking constraint of the 3L-CVRP, while the other packing constraints (see above) are adopted. Another hybrid algorithm for solving the 3L-VRPTW was suggested by Bortfeldt and Homberger (2013). It consists of an evolutionary strategy and two tabu search procedures. Zachariadis, Tarantilis, and Kiranoudis (2012) consider a 3L-VRP with time windows where boxes are stacked on pallets, which in turn are loaded in vehicles. Two hybrid algorithms for the 3L-VRP with backhauls were proposed by Bortfeldt, Hahn, Männel, and Mönch (2015). Both algorithms include a neighborhood search algorithm for routing and a tree search algorithm for packing boxes.

Vehicle routing problems with two-dimensional loading constraints (2L-VRP) are similarly defined as corresponding 3L-VRP but items cannot be stacked on top of each other. In the capacitated vehicle routing problem with 2D loading constraints (2L-CVRP) items and loading spaces of vehicles are rectangles (see Iori, Salazar Gonzalez, & Vigo, 2007). The 2D-CVRP is completed by several packing constraints as LIFO, orientation and weight constraint (see above) that are useful in the 2D-case. An exact approach for solving the 2D-CVRP was proposed by Iori et al. (2007). Metaheuristic methods were suggested, e.g., by Duhamel, Lacomme, Quilliot, and Toussaint (2011), Fuellerer, Doerner, Hartl, and Iori (2009), Gendreau, Iori, Laporte, and Martello (2007) and Wei, Zhang, Zhang, and Lim (2015). Other 2L-VRPs were also investigated, for example the 2L-VRP with time windows (see Khebbache-Hadji, Prins, Yalaoui, & Reghioui, 2013) and the 2L-VRP with heterogeneous fleet (see Leung, Zhang, Zhang, Hua, & Lim, 2013).

2.3. *Pickup and delivery problems with loading constraints*

In recent years, the PDP has been extended by loading constraints in several ways in order to reflect different scenarios occurring in practice. In other cases an extension of PDP to 3L-PDP seems obvious.

Some papers study variants of the 1D-PDP that are extended by loading constraints. Cordeau, Iori, Laporte, and Salazar González (2010) investigate the Traveling Salesman Problem with Pickup and Delivery (TSPPD) with LIFO loading (TSPPDL). Clearly, the TSPPD is a PDP with a single vehicle. In the TSPPDL goods are picked up only at the rear of the vehicle and the LIFO constraint requires that a delivery is only possible if corresponding goods are currently at the rear. Petersen and Madsen (2009) deal with the Double Traveling Salesman Problem with Multiple Stacks (DTSPMS). All pickups and afterwards all deliveries are carried out in two different routes and each stack must observe the LIFO constraint. Coté, Gendreau, and Potvin (2012) examine the single Pickup and Delivery Problem with Multiple Stacks (1-PDPMS). Again, the loading and unloading in each stack must observe a LIFO constraint.

Zachariadis, Tarantilis, and Kiranoudis (2013) deal with the pickup and delivery routing problem with time windows and pallet loading (PDRP-TWP) that is closely related to 3L-PDP. In the PDRP-TWP vehicles have to service plain delivery requests (starting from a central depot) as well as pickup and delivery requests. Products, packaged in boxes, are to be loaded onto pallets which are in turn to be loaded in vehicles. Zachariadis et al. report on an application occurring in the daily inventory management of computer and electronic chain stores. Deliveries from a central depot are often completed by direct transports between pairs of retailers. These transports may happen when goods run out of stock at some retailers while others have excess inventories. The authors stress the necessity to use an integrated routing and loading model in order to guarantee the practicability of routes.

Likewise, Ropke and Pisinger (2006) solve a pickup and delivery problem for a Danish food manufacturer where goods and basic materials are to be transported between facilities of a company. Analogous applications may happen when machine parts, semi-finished products, goods, consumable supplies etc. are transferred between pairs of sites of a company or organization within a restricted urban area and at least in some cases it could be advantageous to apply the 3L-PDP model.

Xu et al. (2003) deal with a practical PDP faced by U.S. logistics firms. The problem has numerous side constraints, e.g. regarding the compatibility of shipped goods and driver regulations. The authors solved large instances with up to 500 requests distributed in large areas. Shipped goods are for example machinery and packaged food. Again, an extension of the model to 3L-PDP is obvious.

Romero, Sheremetov, and Soriano (2007) study a practical PDP that is encountered in helicopter offshore crew transportation of an oil and gas company. Hundreds of employees of the company have to be transported each day across platforms before and after their shift. Since also luggage and other kinds of cargo are to be shipped, a part of the problem could be tackled as 3L-PDP.

Another area of PDP applications can be found in the design of tramp shipping routes (see Brønmo, Christiansen, Fagerholt, & Nygreen, 2007). If cargoes in boxes are to be transported from loading to unloading ports the 3L-PDP model could be applied.

Last not least great courier and parcel service providers offer their customers door to door shipment of bulky goods, parcels of even large dimensions, pieces of furniture etc. If these transports are to be carried out within a urban area or smaller region they can be organized following a PDP or 3L-PDP model (cf. Berbeglia et al., 2007).

Up to now the 2L-PDP was only covered by Malapert, Guéret, Jussien, Langevin, and Rousseau (2008). They develop a constraint programming approach for the loading aspects of the problem but do not report any numerical results. The PDP with 3D loading constraints was only tackled by Bartók and Imreh (2011). They describe a local search heuristic for solving a PDP variant that, however, does neglect the LIFO constraint. Since there are several potential applications, it seems timely to tackle the 3L-PDP in a more extensive fashion.

3. The 3L-PDP and some of its variants

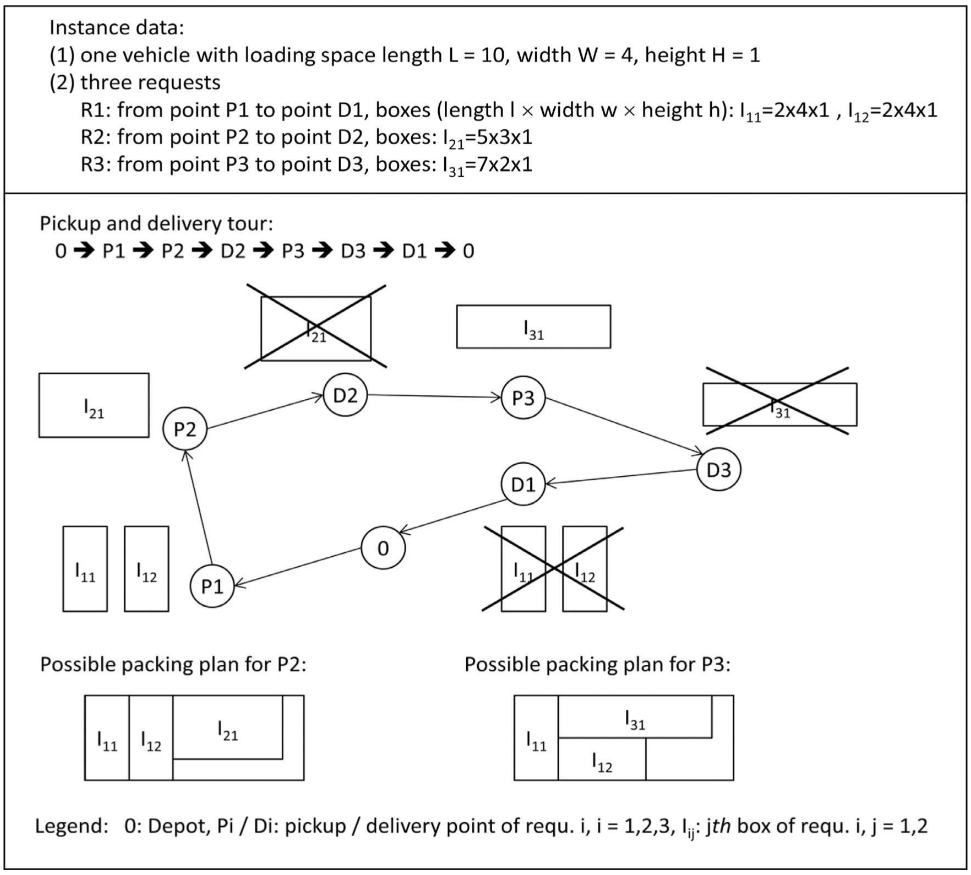
Before giving a more formal definition of the 3L-PDP, we will discuss and illustrate crucial points of this problem.

3.1. *Crucial features of the 3L-PDP*

As in the classical PDP, a number of requests have to be transported from a pickup point to a delivery point by means of homogeneous vehicles. However, in the 3L-PDP, the demands consist of sets of boxes and they are sent in 3D loading spaces of the vehicles.

We assume that all vehicles are rear-loaded, i.e. the goods are loaded and unloaded at the rear exclusively by movements in length direction of the vehicle (cf. Fig. 3). Lifting boxes or moving them in width direction is not permitted in the loading or unloading operation.

At the same time, we want to avoid any *re*loading effort, that is any temporary or permanent repositioning and rotating of boxes *after* loading and *before* unloading. There are different practical reasons to forbid reloading of goods during a pickup and delivery tour. Absence of manpower, tight working time, lack of


Instance data:
(1) one vehicle with loading space length L = 10, width W = 4, height H = 1
(2) three requests
R1: from point P1 to point D1, boxes (length l × width w × height h): I_{11}=2x4x1 , I_{12}=2x4x1
R2: from point P2 to point D2, boxes: I_{21}=5x3x1
R3: from point P3 to point D3, boxes: I_{31}=7x2x1

Pickup and delivery tour:
0 ➔ P1 ➔ P2 ➔ D2 ➔ P3 ➔ D3 ➔ D1 ➔ 0

Legend: 0: Depot, Pi / Di: pickup / delivery point of requ. i, i = 1,2,3, I_{ij}: *jth* box of requ. i, j = 1,2

Fig. 1. A 3L-PDP instance with a feasible solution (view from above).

equipment and shortage of space at customer sites are some of them. Moreover, the goods might be extra heavy, fragile or even hazardous.

Thus, the question arises, which conditions a pickup and delivery tour and the corresponding packing of boxes must observe to rule out any reloading effort.

The first condition is the request sequence (RS) constraint at delivery points that is well-known as last-in-first-out (LIFO) constraint from the 3L-CVRP. At a delivery point the RS constraint requires that between a box A to be unloaded and the rear of the vehicle no box B is situated that needs to be unloaded later. Likewise, a box B to be unloaded later must not lie above box A. Otherwise box B has to be reloaded before box A can be unloaded by a pure movement in length direction.

As also pickup points occur in a pickup and delivery tour, we must have also a RS constraint to exclude reloading of goods at pickup points. At a pickup point the RS constraint requires that between the position of a box A just loaded and the rear of the vehicle or above box A no box B is situated that was loaded at an earlier pickup point. Again, otherwise a reloading of box B would be inevitable.

It is an essential feature of 3L-PDP that the RS constraints for delivery and pickup points are not sufficient to rule out any reloading effort. To understand this fact, we consider a simple 3L-PDP instance and a corresponding solution with one route and appropriate packing plans (see Fig. 1).

It is evident that the route and the appropriate packing plans represent a feasible solution for the 3L-PDP instance given in Fig. 1. In particular, the RS constraint in both variants is observed. Nevertheless, we have to state some reloading effort since box I_{12} of request 1 is rotated at the pickup point of request 3. Moreover, the given route could not be implemented if this reloading operation would not be done. Obviously, it is impossible to find two packing plans for the pickup points *P2* and *P3* such that the boxes of request 1 are located at the same positions in both plans. That is, the boxes of request 1 have to be reloaded at *P3* as otherwise the boxes of requests 1 and 3 could not be stowed together.

It is a specific attribute of the 3L-PDP that in a route boxes of a request *A* can be generally transported and packed for a part of the route together with boxes of a request *B* and for another part of the route together with boxes of a request *C* (and not with the boxes of *B*) etc. In the above example the boxes of request 1 are transported first together with the boxes of request 2 and afterwards together with the boxes of request 3.

If packing plans are generated for the different partial routes in which the boxes of request *A* are “on board”, these boxes will generally occupy different places. To exclude a change of placements without fail, i.e. to rule out a reloading effort, we have to introduce a new constraint, the reloading ban: The placement of any box (including the position of a reference corner and the spatial orientation of the box) must not be changed after the box was loaded and before the box is unloaded.

Routing pattern with sub-patterns with maximal two requests:

$0 \Rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow D1 \Rightarrow P3 \rightarrow P4 \rightarrow D4 \rightarrow D3 \Rightarrow 0$

Routing pattern with sub-patterns with maximal three requests:

$0 \Rightarrow P1 \rightarrow P2 \rightarrow P3 \rightarrow D3 \rightarrow D2 \rightarrow D1 \Rightarrow P4 \rightarrow P5 \rightarrow D5 \rightarrow D4 \Rightarrow 0$

Route that does not follow a routing pattern:

$0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow P3 \rightarrow D3 \rightarrow D1 \rightarrow 0$

Route that does not follow a routing pattern:

$0 \rightarrow P1 \rightarrow P2 \rightarrow D1 \rightarrow D2 \rightarrow 0$

Legend: 0: Depot, Pi / Di: pickup / delivery point of request i, i = 1,…,5, $\Rightarrow$: sub-pattern before is finished

Fig. 2. Examples of routing patterns and routes that do not correspond to routing patterns.

Table 2
Five 3L-PDP variants (y: yes, n: no, a: automatically).

#	RS pickup	RS delivery	Reloading ban	Independent partial routes	Reloading effort	Travel distance
1	y	n	n	n	High	Very low
2	y	y	n	n	Medium	Low
3	y	n	y	n	Medium	Low
4	y	y	y	n	Zero	Medium
5	y	a	a	y	Zero	High

We conclude that reloading effort for the 3L-PDP can only be avoided unerringly if the RS constraints for delivery and pickup points and the reloading ban (as defined) are required at the same time.

However, to meet the reloading ban, we have to find routes for a 3L-PDP instance where each route is completed by a series of interrelated packing plans. In the above example, two interrelated plans are necessary: the first one must contain placements of boxes of requests 1 and 2, the second one must include placements of the boxes of requests 1 and 3. To meet the reloading ban, the placement of boxes of request 1 must be the same in the first and the second plan (making the plans interrelated). The specification of a packing procedure that is able to determine interrelated packing plans for greater routes in short running times presents a fairly difficult task. Therefore, we first look for a simplified 3L-PDP variant that allows us to avoid specifying a packing algorithm for interrelated packing plans.

Instead, we are going to eliminate any reloading effort within a route by means of routing patterns that ensure that the boxes of any request must not be stored together with the boxes of different requests in different partial routes.

The idea of a routing pattern for the 3L-PDP is quite simple. It consists of a series of sub-patterns. Each sub-pattern is a sequence of m ($m \geq 1$) pickup points followed by the corresponding delivery points in inverse order. In Fig. 2 two routing patterns and two routes that do *not* correspond to a routing pattern are shown.

If a route follows a routing pattern, the loading space will become empty again each time after a sub-pattern is finished. Hence, the packing plans that are needed for subsequent sub-patterns (or partial routes) are independent of each other. The boxes of a request have to be stowed together only with boxes of requests of the same sub-pattern. Thus, only one packing plan per request is needed and there is no need to reload the boxes of any request.

If all routes of a solution of a 3L-PDP instance follow a routing pattern in the above sense, we will say that the independent partial routes (IPR) constraint holds.

We are now ready to present a spectrum of five 3L-PDP variants (see Table 2). We always require the RS constraint at pickup points. The 3L-PDP variants are defined by means of the RS constraint for delivery points, the reloading ban and the independent partial routes constraint. For each variant and each constraint the entry is “y”, if the constraint is required and “n” if not. In case the IPR condition and the RS constraint at pickup points is required, RS constraint at delivery points and reloading ban are automatically satisfied (see Section 4.3); this is marked by entry “a”.

Provided none of the three defining constraints must be met, a high reloading effort is to be expected and the total travel distance will be very low. In case only the RS constraint at delivery points holds the reloading effort will be medium while the total travel distance will be low. The same applies if only the reloading ban is required while the RS constraint is not to be observed. For the other variants the reloading effort is zero and the total travel distance is relatively large. Provided the reloading effort is ruled out by the independent partial routes condition the total travel distance is higher as this constraint restricts the solution space stronger than the reloading ban.

We will deal with problem variants 1, 2 and 5 in this paper while variants 3 and 4 are left for a future paper. The 3L-PDP variants and corresponding algorithms have to be compared in terms of total travel distance as well as reloading effort.

Further routing and packing constraints are taken into account. As in the 1D-PDP, we limit the number of routes by a given number of vehicles v_{max} (assuming that each vehicle performs only one route). Also the route length is limited explicitly since the capacity of vehicles does not force a limited route length in a pickup and delivery mode. As usual for 3L-VRP, we extend the problem formulation by some packing constraints introduced by Gendreau et al. (2006), namely a weight constraint, an orientation constraint, a support constraint and a fragility constraint. This procedure is beneficial since it facilitates numerical comparisons with other 3L-VRPs. All six aforementioned constraints are included in all problem variants listed in Table 2.

3.2. Problem definition

Now we describe the 3L-PDP more formally. We are given n requests each consisting of a pickup point i, a delivery point $n+i$ and a set I_i of goods that are to be transported from i to $n+i$ (i 1,…,n). There are v_{max} identical vehicles, originally located at the single depot (denoted by 0), with a rectangular loading space with length L, width W and height H. Let $V = \{0,1,\dots,n,n+1,\dots,2n\}$ be the set of all nodes, i.e. pickup and delivery points including the depot. Let E be a set of undirected edges (i,j) that connect all node pairs ($0 \leq i$, $j \leq 2n$, $i \neq j$) and let $G = (V, E)$ be the resulting graph. Let travel costs c_{ij} ($c_{ij} \geq 0$) be assigned to each edge (i,j) and let the travel costs be symmetric, i.e. $c_{ij} = c_{ji}$ ($0 \leq i$, $j \leq 2n$, $i \neq j$). Set I_i includes m_i rectangular pieces (boxes) I_{ik} and box I_{ik} has the length l_{ik}, the width w_{ik} and the height h_{ik} ($i = 1,\dots,n$, $k = 1,\dots,m_i$).

The loading space of each vehicle is embedded in the first octant of a Cartesian coordinate system in such a way that the length, width and height of the loading space lie parallel to the x, y, and z axes. The placement of box I_{ik} in a loading space is given by the coordinates x_{ik}, y_{ik}, and z_{ik} of the corner of the box closest to the

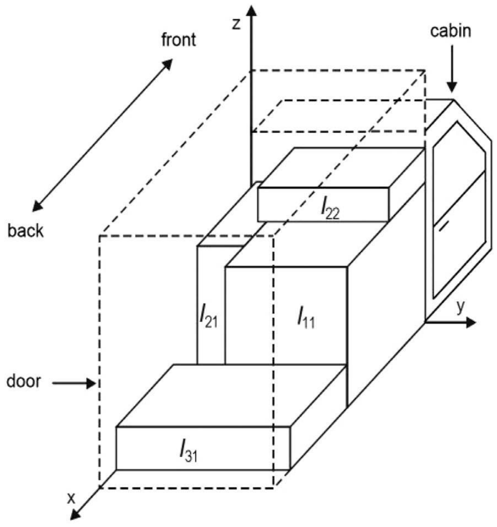


Fig. 3. A loading space with placed boxes.

origin of the coordinates system; in addition, an orientation index o_{ik} indicates which of the possible spatial orientations is selected ($i = 1,...,n$, $k = 1,...,m_i$). A spatial orientation of a box is given by a one-to-one mapping of the three box dimensions and the three coordinate directions.

A packing plan P for a loading space comprises one or more placements and is regarded as feasible if the following three conditions hold: (FP1) each placed box lies completely within the loading space; (FP2) any two boxes that are placed in the same truck loading space do not overlap; (FP3) each placed box lies parallel to the surface areas of the loading space. Fig. 3 shows a loading space with placed boxes. Each vehicle is loaded and unloaded at the rear and empty at the beginning of a route.

A feasible route R is a sequence of $2p+2$ nodes ($p \geq 1$) that starts and ends at the depot. R should include the pickup and delivery points of p different (among the n given) requests and each pickup point must precede the delivery point of the same request. A solution of the 3L-PDP is a set of v sequences (R_l, $P_{l,1}$,…,$P_{l,2pl}$), where R_l is a route and $P_{l,q}$ is a packing plan ($l = 1,…,v$, $q = 1,…,2p_l$, p_l denotes the number of requests of route l).

$P_{l,q}$ represents the packing pattern of route l after having visited its $(q+1)$th node, i.e. after some boxes were loaded or unloaded at the $(q+1)$th node of route l.

To be feasible, a solution must observe the following three conditions: (F1) all routes R_l and packing plans $P_{l,q}$ are feasible ($l = 1,…,v$, $q = 1,…,2p_l$); (F2) the pickup point and the delivery point of each request occurs once in one route R_l ($l = 1,…,v$); (F3) the packing plan $P_{l,q}$ for a route R_l and its $(q+1)$th node contains exactly placements for those boxes which are to be loaded but not (yet) to be unloaded at the first $q+1$ nodes of the route.

In addition, the following routing and packing constraints are to be satisfied optionally:

(C1) *RS constraint for pickup points*: A packed box b of a certain request is said to be in unloading position if there is no packed box b' of another request between b and the rear of the vehicle or above box b (cf. Fig. 3). If the $(q+1)$th node of route l is a pickup point, then all boxes to be loaded there must be in unloading position in the packing plan $P_{l,q}$, i.e. after loading ($l = 1,…,v$, $q = 1,…,2p_l$).

(C2) *RS constraint for delivery points*: If the $(q+1)$th node is a delivery point, then all boxes to be unloaded there must be in unloading position in the packing plan $P_{l,q\text{-}1}$, i.e. before unloading ($l = 1,…,v$, $q = 1,…,2p_l$). Both RS constraints ensure that all boxes of a given request can be loaded or unloaded exclusively by movements parallel to the longitudinal axis of the loading space of a vehicle and without moving boxes of other requests.

(C3) *Reloading ban*: Each box I_{ik} of request i must not be moved *after* loading and *before* unloading ($i = 1,…,n$, $k = 1,...,m_i$). If the box I_{ik} is loaded at the $(q+1)$th node and unloaded at the $(q'+1)$th node of route l, its placement (x_{ik}, $y_{ik,}$, $z_{ik,}$ o_{ik}) must be the same in the packing plans $P_{l,q}$, $P_{l,q+1}$,…,$P_{l,q'\text{-}1}$ ($i = 1,…,n$, $k = 1,...,m_{i,}$ $l = 1,…,v$, $1 \leq q < q' \leq 2p_l$).

(C4) *Independent partial routes constraint*: Each route R_l ($l = 1,…,v$) follows a routing pattern, i.e. it consists of one or more sub-patterns. A sub-pattern consists of a series of one or more pickup points followed by the corresponding delivery points in inverse order.

(C5) *Weight constraint*: Each box I_{ik} has a positive weight d_{ik} ($i = 1,...,n$, $k = 1,...,m_i$) and the total weight of all boxes in a packing plan $P_{l,q}$ must not exceed a maximum load weight D ($l = 1,...,v$, $q = 1,…,2p_l$).

(C6) *Orientation constraint*: The height dimension of all boxes is fixed, while horizontal 90° turns of boxes are allowed. Thus only two of six values are allowed for the orientation index o_{ik} of a placement ($i = 1,...,n$, $k = 1,...,m_i$).

(C7) *Support constraint*: If a box is not placed on the floor, a certain percentage a of its base area has to be supported by other boxes.

(C8) *Stacking constraint*: A fragility attribute f_{ik} ($i = 1,...,n$, $k = 1,...,m_i$) is assigned to each box. If a box is fragile ($f_{ik} = 1$), only other fragile boxes may be placed on its top surface, whereas both fragile and non-fragile boxes may be stacked on a non-fragile box ($f_{ik} = 0$).

(C9) *Route length constraint*: The total distance of a route must not exceed a specified maximum d_{max}. This constraint can also be understood as a route duration constraint if the vehicle velocity is set to a constant.

(C10) *Route number constraint*: The number of routes v must not exceed the number of vehicles v_{max}.

Finally, the 3L-PDP consists of determining a feasible solution that meets some of the constraints (C1)–(C10) and minimizes the total travel distance of all routes. More precisely, we consider the variants of 3L-PDP as specified above and require constraints (C1)–(C4) in accordance to Table 2. The constraints (C5)–(C10) are stipulated for each of the five variants of the 3L-PDP.

4. A hybrid algorithm for the 3L-PDP

In the sequel, we describe a hybrid algorithm for the 3L-PDP consisting of two separate procedures for routing and packing. The routing procedure is derived from the adaptive LNS (ALNS) heuristic for solving the PDPTW by Ropke and Pisinger (2006). Boxes are loaded into vehicles by the tree search 3D packing algorithm by Bortfeldt (2012). In the following description, emphasis is laid on the integration of routing and packing and the economical realization of packing checks.

4.1. Routing procedure

The routing procedure is roughly outlined in Fig. 4. First, an initial solution is constructed. Afterwards, an iterative neighborhood search is carried out until a time limit is exceeded. Within each iteration, a number ξ of requests to be removed and reinserted

```
**3l_pdp_lns** (**in**: problem data, parameters, **out**: best solution $s_{best}$)
    construct initial solution $s_{curr}$ and set $s_{best} := s_{curr}$
    **while** stopping criterion is not met **do**
        select number of requests to be removed $\xi$
        select removal heuristic Rh and insertion heuristic Ih
        determine next solution: $s_{next} := Ih(Rh(s_{curr}, \xi))$
        check acceptance of $s_{next}$
        **if** $s_{next}$ is accepted **then**
            $s_{curr} := s_{next}$
            **if** $(f(s_{curr}) < f(s_{best}))$ **then** $s_{best} := s_{curr}$ **endif**
        **endif**
    **endwhile**
**end**.
```

Fig. 4. LNS-based routing algorithm for the 3L-PDP.

Table 3
Removal and insertion heuristics of the LNS heuristic for 3L-PDP.

Heuristic	Description
Random removal Rh_R	Removes iteratively requests that are selected at random.
Shaw removal Rh_S	Removes iteratively requests that are related in terms of location and weight.
Worst removal Rh_W	Removes iteratively a request whose removal leads to the largest cost (total travel distance) reduction.
Tour removal Rh_T	Removes all requests from a randomly chosen route. If less than ξ requests are removed in this way, further requests will be removed with Shaw removal.
Greedy insertion Ih_G	Inserts iteratively requests into the solution such that the increase of the cost function is minimal.
Regret-2 insertion Ih_{R2}	Inserts iteratively requests into the solution such that the gap in the cost function between inserting the request into its best and its second best route is maximal.
Regret-3 insertion Ih_{R3}	Inserts iteratively requests into the solution such that the sum of two gaps in the cost function is maximal. The first gap results from inserting the request into its best and its second best route, while the second gap results from inserting the request into its best and its third best route.

in the solution is selected randomly. Several removal and insertion heuristics are available. Among them, one removal and one insertion heuristic are selected randomly per iteration. The next solution is generated by the selected heuristics *Rh* and *Ih* according to $s_{next} := Ih(Rh(s_{curr}, \xi))$. If s_{next} is accepted in a dedicated test, it becomes the new current solution s_{curr} and the best solution s_{best} is updated if necessary. Otherwise, the initial solution of the next iteration s_{curr} remains unchanged.

The acceptance test follows the well-known simulated annealing rule and according to this, the search is embedded in an annealing process with a geometric cooling schedule. Differently to the original adaptive LNS, the selection probabilities for the removal and insertion heuristics are fix; i.e. a pure LNS is performed. Moreover, no noise term is applied to the objective function.

Since the number of vehicles is limited, it may happen that the initial solution or a later generated solution is incomplete, i.e. some requests are missing. To cope with this situation, the concept of a virtual request bank is used as in the original ALNS heuristic. The objective function is defined as the sum $f(s) = ttd(s) + M \cdot nmc(s)$, where s is a given solution, *ttd* stands for its total travel distance, *nmc* is the number of missing requests and M is a sufficiently large constant. By this definition, solutions with less missing requests are always preferred.

The removal and insertion heuristics are basically adopted from the original ALNS heuristic and briefly summarized in Table 3. Within the Shaw removal, the relatedness of requests is expressed by means of two factors, namely the locations of their pickup and delivery points and the weights of their item sets (see Shaw, 1998). Hence, the relatedness of the two requests i and j is calculated by the blended index $r(i,j) = w_{r1}(c'_{ij} + c'_{i+n,j+n}) + w_{r2}|v'_i - v'_j|$, $1 \le i < j \le n$, where c'_{ij}, $c'_{i+n,j+n}$ and v'_i lie in the interval [0,1]; c'_{ij} ($c'_{i+n,j+n}$) denotes the normalized distance between the pickup points i and j (the delivery points $i+n$ and $j+n$); v'_i denotes the normalized weight of request i. The weights w_{rp} ($p = 1$, 2) allow for a different weighting of the distances and weights difference.

The Tour removal has been added in order to drive the search into regions where feasible solutions with less tours can be found. Although the minimization of the number of tours is not an explicit goal, this procedure can be helpful to identify high-quality solutions in terms of total travel distance.

The insertion heuristics are based on insertion moves and the concept of insertion cost. An insertion move is specified by four parameters: i denotes the inserted request ($1 \le i \le n$), k indicates the route in which i is inserted ($1 \le k \le v$), β and ε are the positions where the pickup and delivery point of request i are inserted into route k ($2 \le \beta \le 2p(k)+2$, $3 \le \varepsilon \le 2p(k)+3$, $\beta < \varepsilon$); $p(k)$ is the number of requests that currently belong to route k. The insertion cost $\Delta f(i,k,\beta,\varepsilon)$ of an insertion move stands for the increase of the objective function value if the move is implemented. Only those insertion moves are admitted that do not violate the weight constraint (C5) and the route length constraint (C9). For a given request and a route k, having currently $p(k)$ requests, at most $(2p(k)+1)(2p(k)+2)/2$ feasible combinations for the index pair (β, ε) are available. Fig. 5 shows two variants of inserting a request into a route.

For the insertion variant $(\beta,\varepsilon) = (3,7)$ (on the right) the insertion cost $\Delta f(i,k,\beta,\varepsilon)$ results by adding the distances $P1 \rightarrow P3$, $P3 \rightarrow P2$, $D2 \rightarrow D3$ and $D3 \rightarrow 0$ and subtracting then $P1 \rightarrow P2$ und $D2 \rightarrow 0$.

The insertion cost of a request i into a route k is specified as $\Delta f(i,k) = \min_{\beta,\varepsilon} \Delta f(i,k,\beta,\varepsilon)$, i. e. $\Delta f(i,k)$ is given by the cheapest insertion move of request i into route k. The insertion heuristics perform several iterations. The Greedy insertion Ih_G selects in each iteration the (request, route)-pair (i_0, k_0) for insertion which minimizes the insertion cost ($\Delta f(i^0, k^0) = \min_{i,k} \Delta f(i,k)$).

The Regret-2 insertion Ih_{R2} heuristic selects per iteration the request i_0 that maximizes the regret value $\rho_2(i) = \Delta f(i, k_2(i)) - \Delta f(i, k_1(i))$; k_1 and k_2 ($k_1 \neq k_2$) are those routes in which request i can be inserted with smallest and second smallest insertion cost ($\Delta f(i, k_1(i)) \le \Delta f(i, k_2(i)) \le \Delta f(i,k) \quad \forall k, k \neq k_1, k \neq k_2$), respectively. Finally, we mention that the regret value for the

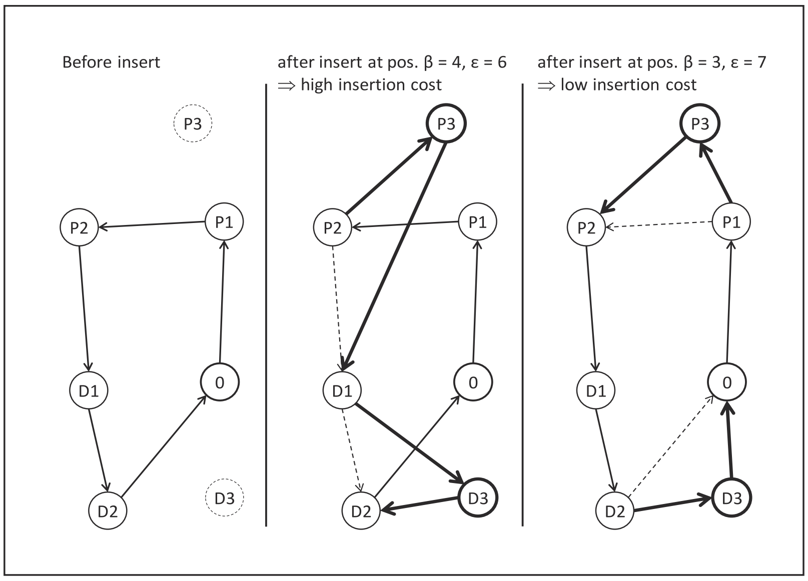


Fig. 5. Insertion of a request into a route (bold lines: added edges, dotted lines: removed edges).

Regret-3 insertion Ih_{R3} is calculated according $\rho_3(i) = \Delta f(i, k_2(i)) - \Delta f(i, k_1(i)) + \Delta f(i, k_3(i)) - \Delta f(i, k_1(i))$.

The initial solution is constructed by means of the Regret-2 insertion heuristic starting with an empty solution.

4.2. Integration of routing and packing

To provide feasible packing plans for routes of solutions 3D packing checks are performed that are integrated in two parts of the routing procedure.

Let a new solution (s_{next}) be generated from an old one (s_{curr}) by means of a removal heuristic *Rh* and an insertion heuristic *Ih* according to $s_{next} := Ih(Rh(s_{curr}, \xi))$ and consider a route of s_{curr}. If *Rh* and *Ih* are applied to the route two cases can occur. In general, some requests (i.e. pairs of a pickup and a delivery point) of the route are removed and some new requests are reinserted. It might also occur that only old requests are removed from the route without inserting new ones. In the former case, it will suffice to integrate packing checks in the insertion heuristic that is applied after the removal heuristic. In the latter case, it will be mostly possible to store the boxes of the remaining requests at all remaining sites of a route in a feasible way, too. Therefore, packing checks are integrated in insertion heuristics exclusively (and not in removal heuristics) and these checks are called insertion packing checks.

However, sometimes the boxes of a given set can be stored in the loading space of a route in a feasible way, while this is no longer the case after some of the boxes were removed. Such a situation may occur, e.g. if a box that is needed to provide sufficient support for another fragile box was removed. To cope with these cases, packing checks will also be applied to all routes of a solution s_{next} within the acceptance test of s_{next} (see Fig. 4) and these checks are called acceptance packing checks. Especially all routes that did result earlier by a pure removing of requests are checked. If there is at least one site for which no feasible packing plan can be provided, the solution s_{next} will be discarded and the search continues with the last accepted solution. This measure prevents that an accepted solution ever includes an infeasible route in terms of packing.

Subsequently, the integration of insertion packing checks is shown by means of the Greedy insertion heuristic.

In Fig. 6 the Greedy insertion heuristic is shown in detail. It is based on the procedure select_best_insertions that performs the packing checks and is shown in Fig. 7. The Greedy insertion heuristic takes an incomplete solution, a set of missing requests and the best solution so far as input values. In each loop cycle the best (minimum cost) insertion is determined, related to the requests still missing and implemented before the set of missing requests is updated. The procedure select_best_insertions is used to deliver the best insertion for a given request.

In each cycle the number of still missing requests nmr_{wi} for which no feasible insertion was found at all is counted. If in any cycle nmr_{wi} is greater than the number of missing requests $nmr(s_{best})$ in the best solution found so far, then a further computation is probably useless and the heuristic will end. Otherwise, a solution is provided in the end that has no more missing requests than the best solution so far or is even a feasible and complete solution.

The procedure select_best_insertions is organized in two parts. In the first part (*for*-loop) all potential insertions of a given request *rq* into any route of a given solution *s* are provided. Each insertion must be feasible in terms of route length (C9), route number (C10) and weight (C5). The minimum cost insertions of all routes are collected in a list I_{cand}.

In the second part (*while*-loop), the insertions of I_{cand} are examined by ascending costs. In each cycle the currently minimum cost insertion ins_{best} undergoes a 3D packing check, i.e. the insertion ins_{best} is applied to its route and the route is then checked in terms of the constraints (C1) and (C6)–(C8). If the outcome is positive, insertion ins_{best} is included into the set of best insertions I_{best} (and removed in I_{cand}).

Otherwise the next cheapest insertion for the route of ins_{best} (if any) will replace ins_{best} in list I_{cand}. The procedure ends if I_{best} has enough (n_{ins}) insertions or if I_{cand} is empty. Any two insertions in I_{best} belong to different routes.

Two features of the procedure select_best_insertions should be stressed. First, one-dimensional checks are made before 3D

```
greedy_insertion (in: set of missing requests Rm, $s_{best}$, inout: solution s)
    repeat
        min_cost := $\infty$
        $nmr_{wi}$ := 0                                   // no. missing requests without insertion
        for all rq $\in$ Rm do
            $I_{best}$(rq) := select_best_insertions(s,rq,1) // set $I_{best}$(rq) receives best rq-insertion
            if $|I_{best}(rq)|$ = 0 then $nmr_{wi}$ := $nmr_{wi}$ + 1   // request without insertion
            else
                if cost of best rq-insertion < min_cost then
                    $rq_{ins}$ := rq; min_cost := cost of best rq-insertion endif
            endif
        endfor
        if $nmr_{wi}$ > nmr($s_{best}$) then return endif          // no useful solution
        if $nmr_{wi}$ <|Rm| then update s by insertion in $I_{best}(rq_{ins})$; Rm := Rm \ {$rq_{ins}$} endif
    until (|Rm| = 0 or $nmr_{wi}$ =|Rm|)
end.
```

Fig. 6. Greedy insertion heuristic.

```
select_best_insertions (in: solution s, request rq, no. of required insertions $n_{ins}$,
                        out: set of best rq-insertions $I_{best}$)
    $I_{best}$ := Ø; list of insertion candidates $I_{cand}$ := Ø
    for all routes r of solution s do
        $I_{route}$(r) := set of all 1D-feasible insertions of rq in route r
        sort $I_{route}$(r) by ascending cost
        if $|I_{route}(r)|$ > 0 then $I_{cand}$ := $I_{cand} \cup \{I_{route}(r)(1)\}$ endif     // add first insertion of $I_{route}$(r)
    endfor
    while $|I_{best}|$ < $n_{ins}$ and $|I_{cand}|$ > 0 do
        sort $I_{cand}$ by ascending cost
        best insertion $ins_{best}$ := $I_{cand}$(1); $I_{cand}$ := $I_{cand}$ \ {$ins_{best}$}
        perform 3D packing check of insertion $ins_{best}$, i.e. of the resulting route
        if 3D packing check of $ins_{best}$ successful then
            $I_{best}$ := $I_{best} \cup \{ins_{best}\}$    // next best insertion found
        else  r := route of $ins_{best}$
            $I_{route}$(r) := $I_{route}$(r) \ {$ins_{best}$} // remove $ins_{best}$
            if $|I_{route}(r)|$ > 0 then $I_{cand}$ := $I_{cand} \cup \{I_{route}(r)(1)\}$ endif // add (new) first insertion
        endif
    endwhile
end.
```

Fig. 7. Procedure select_best_insertions with packing check.

packing checks are carried out. Second, all possible insertions are first evaluated and sorted by cost *before* the “expensive” packing checks are made. By this technique, called “evaluating first, packing second”, the packing effort is kept low since the packing checks can be aborted each time after few (3D-)feasible insertions have been detected.

The procedure select_best_insertions is also used for the Regret-2 and Regret-3 insertion heuristics. In every cycle these heuristics retrieve the best two (three) insertions from select_best_insertions for all requests currently not contained in the (incomplete) solution to calculate the requests regret values. For each request, these two (three) insertions must belong to different routes. In each cycle the request with maximal regret value is inserted into the solution, i.e. these requests' best insertion is implemented. For details about the Regret-2 and Regret-3 insertion heuristic see Bortfeldt et al. (2015).

Our implementation of the LNS routing procedure can also be applied to the (1D-)PDP. In this situation, the 3D packing test is omitted and only the route length (C9), route number (C10), and weight constraint (C5) are checked.

4.3. The concept of packing checks

The insertion packing checks and the acceptance packing checks are now explained in detail. Basically, for each route of a solution and each site visited in this route a feasible packing plan has to be provided. The plan must stow all boxes that are already loaded and not yet unloaded after the visit of this site. Now we ask whether existing feasible packing plans for selected sites of a route guarantee the existence of feasible packing plans for other sites. It turns out that this is the case at least if additional requirements hold that are based on the required constraints of the 3L-PDP variants dealt with in this paper, i.e. the variants 1, 2 and 5 (see Table 2). By using these additional requirements we want to reduce the effort spent for packing checks. We first deal with the 3L-PDP variants 2 and 5 and afterwards with variant 1.

We define a sequence of open pickup points (SOPP) as a sequence of pickup points within a route of a 3L-PDP solution with following characteristics: (i) the last point of the sequence is followed by a delivery point in the route; (ii) the sequence contains exactly all pickup points of the route whose delivery points lie behind the last sequence point. Examples of SOPPs can be found in Fig. 8 below.

Let m_2 ($m_2 \geq 1$) be the number of consecutive pickup points lying at the end of the SOPP. Let m_1 ($m_1 \geq 0$) be the number of pickup points that are separated from the last m_2 pickup points by at least one delivery point. Then the sequence can be denoted as P_i, $i = 1,\ldots,m_1$, $m_1+1,\ldots,m_1+m_2$ (i.e. P_{m1+m2} is the last point).

We say that a packing plan for pickup point P_{m1+m2} of a SOPP satisfies the cumulative request sequence constraint for pickup

Example for problem variant 2:

Given route

$0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow P3 \rightarrow P4 \rightarrow P5 \rightarrow D5 \rightarrow D4 \rightarrow P6 \rightarrow D6 \rightarrow D3 \rightarrow D1 \rightarrow 0$

Sequence of open pickup points	Packing plan incl. CRS-p constraint to be provided for pickup point	Derived packing plans result for sites
1. $P1 \rightarrow P2$	P2	P1, D2
2. $P1 \rightarrow P3 \rightarrow P4 \rightarrow P5$	P5	P3, P4, D5, D4
3. $P1 \rightarrow P3 \rightarrow P6$	P6	D6, D3, D1

Example for problem variant 5:

Given route

$0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow D1 \rightarrow P3 \rightarrow P4 \rightarrow P5 \rightarrow D5 \rightarrow D4 \rightarrow D3 \rightarrow P6 \rightarrow D6 \rightarrow 0$

Sequence of open pickup points	Packing plan incl. CRS-p constraint to be provided for pickup point	Derived packing plans result for sites
1. $P1 \rightarrow P2$	P2	P1, D2, D1
2. $P3 \rightarrow P4 \rightarrow P5$	P5	P3, P4, D5, D4, D3
3. P6	P6	D6

Legend: 0: Depot, Pi / Di: pickup / delivery point of request i, i = 1,…,6.

Fig. 8. Packing checks in 3L-PDP variants 2 and 5.

points (CRS-p) if the following conditions hold: (i) there are no boxes of a request j (loaded at pickup point P_j) between a box of request i and the rear of the vehicle; (ii) there are no boxes of request j above a box of request i ($i, j = 1,\ldots,m_1+m_2$, $j < i$).

Proposition 1. *Let a feasible plan for pickup point P_{m1+m2} of a SOPP exist that meets the constraints (C6)–(C8) and observes the CRS-p constraint. Then feasible packing plans observing constraints (C1) and (C6)–(C8) do also exist for pickup points P_i ($i = m_1+1,\ldots,m_1+m_2-1$).*

Proof. Clearly, a packing plan for a pickup point that observes the CRS-p constraint also meets the RS constraint for pickup points (C1). If the boxes of the last request m_1+m_2 are removed, a packing plan for pickup point $P_{m1+m2-1}$ results that also meets the CRS-p constraint, hence the (C1) constraint. The plan for $P_{m1+m2-1}$ also satisfies support constraint (C7), as no boxes were removed that could serve for supporting boxes of requests 1 to m_1+m_2-1 (condition (ii) in CRS-p constraint definition). Constraints (C6) and (C8) as well as feasibility conditions (FP1) to (FP3) are trivially met in the plan for $P_{m1+m2-1}$ because they hold in the plan for P_{m1+m2}. For the pickup points P_i ($i = m_1+m_2-2,\ldots, m_1+1$), packing plans can be derived in a similar manner. □

Now we introduce a further constraint with regard to routing called inverse delivery points sequence (IDPS) constraint. A route meets the IDPS constraint if the following condition holds: given any two requests i and j that are transported together at least between two consecutive points of a route; if the pickup point P_i lies before P_j, then the delivery point D_i lies behind D_j ($i,j = 1,\ldots,n$, $i < j$).

Proposition 2. *Let a route be given that observes the IDPS constraint and a SOPP within the route. Let a feasible packing plan for the (last) pickup point P_{m1+m2} exist that meets the constraints (C6) to (C8) and satisfies the CRS-p constraint. Then feasible packing plans observing the constraints (C2) and (C6) to (C8) do also exist for the consecutive delivery points following pickup point P_{m1+m2}.*

Proof. The points following P_{m1+m2} must be the delivery points D_{m1+m2}, $D_{m1+m2-1},\ldots$, $D_{m1+m2-\ m3}$ ($0 \le m_3 \le m_1+m_2$) in this order since another set and another order of delivery points would contradict the IDPS constraint. The boxes of request m_1+m_2 are already in unloading position if the vehicle arrives in D_{m1+m2}. After these boxes were unloaded the boxes of request m_1+m_2-1 are in unloading position due to the CRS-p constraint, i.e. constraint (C2) is met in the plan for D_{m1+m2}. Constraints (C6)–(C8) and feasibility conditions (FP1) to (FP3) are verified for the plan for delivery point D_{m1+m2} as in proof of Proposition 1. Feasible packing plans for further consecutive delivery points can be derived similarly. □

The above considerations show that for 3L-PDP variants 2 and 5 it is sufficient to construct feasible packing plans for the last pickup points of all SOPPs of a given route. Feasible packing plans that observe the RS constraints (C1) and (C2) and constraints (C6)–(C8) can in this case be derived for all other pickup points and all delivery points of this route.

However, this claim holds only if two conditions are fulfilled. On the one hand, the constructed packing plans for the last pickup points of SOPPs must observe the CRS-p constraint being stronger than the (C1) constraint. On the other hand, the IDPS constraint for routes must be required. This constraint is included in the independent partial routes constraint (see Section 3.1); hence it holds automatically in problem variant 5. In problem variant 2 the IDPS constraint is substituted for the RS constraint for delivery points (C2). This is possible as the RS constraint for pickup points (C1) and the IDPS constraint result in packing plans for delivery points that meet constraint (C2) as shown in Proposition 2 (this way also the entry “a” in line 5 and column “RS delivery” in Table 2 is justified).

While a packing procedure has to deliver packing plans for all last points of SOPPs in a given route that must observe the CRS-p constraint, the IDPS and IPR constraint are routing constraints that can be checked within procedure select_best_insertions together with other routing constraints.

Acceptance packing checks for problem variants 2 and 5 are carried out exactly as described before. An insertion packing check is performed each time another request is inserted into a route (see Section 3.2). Packing plans are then to be provided only for those last pickup points of SOPPs that lie between the inserted pickup point and the inserted corresponding delivery point since for the other parts of the route feasible packing plans are provided with the unmodified route of the former solution.

The procedure of packing checks for a route in 3L-PDP variants 2 and 5 is illustrated by two examples in Fig. 8. In the first example (variant 2), boxes of some requests, e.g. request 1, are loaded in multiple packing plans with possibly different placements. In the second example (variant 5) each request is stowed only one time due to the IPR constraint. Hence, there is no reloading effort at all. Note that the route observes the IDPS constraint in the first example and the IPR constraint in the second one.

In the 3L-PDP variant 1, the RS constraint for pickup points (C1) has to be observed. Hence, packing checks for all last pickup points of SOPPs are carried out as in problem variants 2 and 5. However, the RS constraint for delivery points (C2) and a fortiori the stronger IDPS constraint is no longer required. Therefore, packing plans for delivery points cannot be derived generally from packing plans for previous pickup points and their existence is no longer guaranteed. Instead, feasible packing plans for all delivery points of a given route must be provided separately. This is done in the following way:

(1) If a vehicle arrives at a delivery point, all boxes of the corresponding request, say *A*, are to be unloaded. Since the RS constraint (C2) is not required, some boxes of requests *B*, *C*, etc. may stand in the way of the *A*-boxes. These are called blocking boxes. We assume that blocking boxes have to be temporarily unloaded.
(2) It can occur that there are no blocking boxes at all, i.e. all *A*-boxes are in unloading position. In this case the packing plan for the current delivery point results from the plan for the former site simply by removing the *A*-boxes. The resulting plan is feasible and observes the constraints (C6)–(C8) if the former plan has these characteristics (see Proposition 2).
(3) Otherwise a new feasible packing plan must be built for the current delivery point. The plan must contain placements for the loaded and not yet unloaded boxes (i.e. no longer placements of *A*-boxes).
(4) In particular, the boxes that were temporarily unloaded must be considered for the new packing plan. We define two policies for doing this. In the first policy (resulting in *subvariant 1A*), the original loading order of all requests is maintained. In the second policy (*subvariant 1B*), the original loading order is modified as the order of requests for which temporarily unloaded boxes do exist is now chosen inverse to the order of corresponding delivery points. Clearly, the second loading order should result in less reloading effort at the following delivery points.
(5) For the specified loading order, the packing heuristic generates a packing plan (if possible) in such a way that constraints (C6)–(C8) and the CRS-p constraint for the given loading order is fulfilled.
(6) For the boxes that have to be temporarily unloaded, a reloading effort is to be calculated in any case. For other boxes, the placements in the packing plan for the current delivery point are compared with the placements in the packing plan of the previous site. Only for boxes with changed placements a reloading effort is calculated (including reloading effort for their blocking boxes if necessary).
(7) Similar differences concerning the scope of checks exist between acceptance packing checks and insertion packing checks as in problem variants 2 and 5 (see above).
(8) The structure of input data of a packing check for a pickup point and for a delivery point is the same. In any case, the loading order of the relevant requests is needed as well as the box set per request.

Packing checks for problem variant 1 are illustrated by an example in Fig. 9 that considers only delivery points. Note that at delivery point D3 boxes of request 2 cannot be blocking boxes because of constraint (C1).

4.4. Packing procedure

For a given loading sequence of requests and the corresponding sets of boxes, the packing procedure tries to determine a complete solution, i.e. a packing plan stowing all given boxes. The generated packing plan is feasible and observes the constraints (C6)–(C8) as well as the CRS-p constraint (including the (C1) constraint). A depth first search is carried out by means of the recursive procedure add_placement shown in Fig. 10. A stowage plan *currentSolution* is transferred and then extended in different variants by one further box placement for each procedure call.

As the search is started, the solution *currentSolution* is set empty, the set *freeBoxes* is filled by all boxes (incl. data concerning loading sequence) and the list *potentialPlacements* is filled by all feasible box placements in the lower left front corner of the loading space $L \times W \times H$ (cf. Fig. 3).

The procedure add_placement checks first whether the current packing check can be aborted. This is done, i.e. all running instances of procedure add_placement are aborted, if *currentSolution* is a complete solution or if the number of calls of add_placement exceeds a given limit *maxApCalls.* The current instance of the procedure is aborted if there is at least one free box without a potential placement, i.e. if a complete solution can no longer be achieved.

Candidates for the next placement are selected from the list *potentialPlacements* and provided in the list *currentPlacements*. All these placements are then tried alternatively. For each placement, the current solution, the set of free boxes, and the list of potential placements are updated accordingly before procedure add_placement is called again. To update the list *potentialPlacements,* all potential placements that can no longer be implemented are removed. Additional potential reference points for new

Example for problem variant 1:

Given route

$0 \rightarrow P1 \rightarrow P2 \rightarrow D1 \rightarrow P3 \rightarrow P4 \rightarrow P5 \rightarrow D3 \rightarrow D4 \rightarrow D5 \rightarrow D2 \rightarrow 0$

Delivery point	New packing plan to be provided	Relevant requests	Request numbers of possibly blocking boxes	Loading order
1. D1	possibly yes	2	2	2
2. D3	possibly yes	2,4,5	4,5	$2 \rightarrow 4 \rightarrow 5$ (1A) $2 \rightarrow 5 \rightarrow 4$ (1B)
3. D4	possibly yes (1A) no (1B)	2,5	5	$2 \rightarrow 5$
4. D5	no			
5. D2	no			

Legend: 0: Depot, i: request no., Pi / Di: pickup / delivery point of request i, i = 1,...,5.

Fig. 9. Packing checks for delivery points in 3L-PDP variant 1.

```
add_placement (in: freeBoxes, potentialPlacements, inout: currentSolution)
if all boxes stowed in currentSolution or number of procedure calls > maxApCalls then
        abort packing check endif
// abort current instance
if there is at least one free box without placement in potentialPlacements then return endif
provide list currentPlacements with potential placements that are currently to be tried
for i := 1 to |currentPlacements| do
        currentSolution' := currentSolution ∪ { currentPlacements(i) }          // add placement to solution
        freeBoxes' := freeBoxes \ { currentPlacements(i).box }                  // update free boxes
        potentialPlacements' := update(potentialPlacements)                     // update potential placements
        add_placement (currentSolution', freeBoxes', potentialPlacements')      // recursive call
endfor
end.
```

Fig. 10. Packing procedure add_placement.

potential placements are determined as extreme points (see Crainic, Perboli, & Tadei, 2008).

The selection of placements currently to be tried among all potential placements is governed by two rules. On the one hand, it is ensured that a vehicle is loaded from front to rear, from bottom to top with lower priority, and from left to right with lowest priority. Hence, placements with smaller *x*-coordinates of the reference corner are preferred, etc. On the other hand, the selection is made taking into account the CRS-p constraint. Placements of boxes are preferred that belong to earlier loaded requests and, therefore, have to be stowed nearer to the cabin. The placement selection is controlled by the integer parameters *maxBoxRankDiff* and *maxRefPoints* where higher parameter values lead to a larger set of currently tried placements (see Bortfeldt, 2012).

All loading sequences of requests that have ever been checked are collected in a cache to further accelerate the search. Whenever a sequence is tested, it is first searched in the cache. To speed up this search, for each request a separate table is established keeping the positions of all stored sequences including this request. Thus, a request sequence is searched by examining only the cache positions of its first request. The packing algorithm is only called if the sequence was not found in the cache and the sequence is then inserted in the cache together with the result of the packing test. Multiple checks of same request sequences are avoided by this procedure.

5. Computational experiments

The computational experiments are organized in two parts. In the first part we examine the 1D variant of the hybrid algorithm (denoted by 1D-LNS) using the well-known PDP instances by Li and Lim (2001) with up to 100 requests. In the second part we test mainly the 3D variant of the algorithm by means of 54 new 3L-PDP instances with up to 100 requests and up to 300 boxes.

The packing procedure is coded in the C++ programming language using Visual Studio 2012 Express, while the LNS scheme is implemented using the Java programming language under Eclipse 3.5.2. Preliminary experiments (in which total run times were varied) demonstrated that the impact of the different developing environments is negligible. All the experiments have been conducted on a PC with Intel Core i5-2500 K (4.0 gigahertz, 16 gigabyte RAM).

Afterwards, the new benchmark instances are introduced and the parameter setting is specified before the computational results are presented and analyzed.

5.1. Benchmark instances for 3L-PDP

To provide a sufficiently large set of 3L-PDP benchmark instances with different characteristics we generate the instances as follows:

Table 4
Overview of the 54 new 3L-PDP benchmark instances.

Number of	2 Boxes per request on average			3 Boxes per request on average			Total
requests	Random	Mixed cluster	Pure cluster	Random	Mixed cluster	Pure cluster	
50	5	5	5	5	5	5	30
75	3	3	3	3	3	3	18
100	1	1	1	1	1	1	6

Table 5
Parameter setting for the LNS routing procedure.

Parameter	Description	Value
r_{min}	Lower bound for no. of removed customers	0.04·n
r_{max}	Upper bound for no. of removed customers	0.4·n
w	Start temperature control parameter	0.005
c	Rate of geometrical cooling	0.9999
$p(Rh_R)$, $p(Rh_S)$	Probability of random/Shaw removal	0.3, 0.4
$p(Rh_W)$, $p(Rh_T)$	Probability of worst/tour removal	0.1, 0.2
$p(Ih_G)$, $p(Ih_{R2})$, $p(Ih_{R3})$	Probability of greedy/regret-2/regret-3 insert	0.1, 0.6, 0.3
w_{r1}, w_{r2}	Weights of relatedness formula for Shaw removal	9,2

Table 6
Parameter setting for packing procedure.

Parameter	Description	Value
maxApCalls	Max. no. of calls to procedure add_placement	3000
maxBoxRankDiff	Max. tolerated rank difference of boxes	2
maxRefPoints	Max. number of admitted reference points	3

Table 7
Computing time in minutes for experiments with (3L-)PDP instances.

Number of requests	1D-test (no packing)	2 Boxes per request on avg.	3 Boxes per request on avg.
50	1	2	5
75	2	4	10
100	4	8	20

- 30 instances with 50 requests, 18 instances with 75 and 6 instances with 100 requests are provided. The average number of boxes per request is two for half and three for the other half of the instances.
- Regarding the distribution of pickup and delivery points of the requests, we distinguish the three variants “Random”, “Mixed cluster” and “Pure cluster”. In variant “Random” the sites are uniformly distributed in a rectangular section of the plane, while they are clustered in the other variants. In variant “Mixed

Table 8
Best solutions for Li-Lim-100 instances (50 requests).

Instance	1D-LNS		Ropke and Pisinger (2006)		Best known solution	
	nv	ttd	nv	ttd	nv	ttd
LC101	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**
LC102	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**
LC103	**9**	**1035.35**	**9**	**1035.35**	**9**	**1035.35**
LC104	**9**	**860.01**	**9**	**860.01**	**9**	**860.01**
LC105	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**
LC106	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**
LC107	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**
LC108	**10**	**826.44**	**10**	**826.44**	**10**	**826.44**
LC109	**9**	**1000.60**	**9**	**1000.60**	**9**	**1000.60**
LC201	**3**	**591.56**	**3**	**591.56**	**3**	**591.56**
LC202	**3**	**591.56**	**3**	**591.56**	**3**	**591.56**
LC203	3	591.17	3	591.17	**3**	**585.56**
LC204	**3**	**590.60**	**3**	**590.60**	**3**	**590.60**
LC205	**3**	**588.88**	**3**	**588.88**	**3**	**588.88**
LC206	**3**	**588.49**	**3**	**588.49**	**3**	**588.49**
LC207	**3**	**588.29**	**3**	**588.29**	**3**	**588.29**
LC208	**3**	**588.32**	**3**	**588.32**	**3**	**588.32**
LR101	**19**	**1650.80**	**19**	**1650.80**	**19**	**1650.80**
LR102	**17**	**1487.57**	**17**	**1487.57**	**17**	**1487.57**
LR103	**13**	**1292.68**	**13**	**1292.68**	**13**	**1292.68**
LR104	**9**	**1013.39**	**9**	**1013.39**	**9**	**1013.39**
LR105	**14**	**1377.11**	**14**	**1377.11**	**14**	**1377.11**
LR106	**12**	**1252.62**	**12**	**1252.62**	**12**	**1252.62**
LR107	**10**	**1111.31**	**10**	**1111.31**	**10**	**1111.31**
LR108	**9**	**968.97**	**9**	**968.97**	**9**	**968.97**
LR109	**11**	**1208.97**	**11**	**1208.96**	**11**	**1208.96**
LR110	**10**	**1159.35**	**10**	**1159.35**	**10**	**1159.35**
LR111	**10**	**1108.90**	**10**	**1108.90**	**10**	**1108.90**
LR112	**9**	**1003.77**	**9**	**1003.77**	**9**	**1003.77**
LR201	**4**	**1253.23**	**4**	**1253.23**	**4**	**1253.23**
LR202	**3**	**1197.67**	**3**	**1197.67**	**3**	**1197.67**
LR203	**3**	**949.40**	**3**	**949.40**	**3**	**949.40**
LR204	**2**	**849.05**	**2**	**849.05**	**2**	**849.05**
LR205	**3**	**1054.02**	**3**	**1054.02**	**3**	**1054.02**
LR206	**3**	**931.63**	**3**	**931.63**	**3**	**931.63**
LR207	**2**	**903.06**	**2**	**903.06**	**2**	**903.06**
LR208	**2**	**734.85**	**2**	**734.85**	**2**	**734.85**
LR209	**3**	**930.59**	**3**	**930.59**	**3**	**930.59**
LR210	**3**	**964.22**	**3**	**964.22**	**3**	**964.22**
LR211	**2**	**911.52**	**2**	**911.52**	**2**	**911.52**
LRC101	**14**	**1708.80**	**14**	**1708.80**	**14**	**1708.80**
LRC102	**12**	**1558.07**	**12**	**1558.07**	**12**	**1558.07**
LRC103	**11**	**1258.74**	**11**	**1258.74**	**11**	**1258.74**
LRC104	**10**	**1128.40**	**10**	**1128.40**	**10**	**1128.40**
LRC105	**13**	**1637.62**	**13**	**1637.62**	**13**	**1637.62**
LRC106	**11**	**1424.73**	**11**	**1424.73**	**11**	**1424.73**
LRC107	**11**	**1230.15**	**11**	**1230.14**	**11**	**1230.14**
LC101	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**
LC102	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**
LC103	**9**	**1035.35**	**9**	**1035.35**	**9**	**1035.35**
LC104	**9**	**860.01**	**9**	**860.01**	**9**	**860.01**
LC105	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**
LC106	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**
LC107	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**
LC108	**10**	**826.44**	**10**	**826.44**	**10**	**826.44**
LC109	**9**	**1000.60**	**9**	**1000.60**	**9**	**1000.60**
Average		**1036.78**		**1036.68**		**1036.78**

Table 9
Best solution for Li-Lim-200 instances (100 requests).

Instance	1D-LNS		Ropke and Pisinger (2006)		Best known solution	
	nvt	ttd	nv	ttd	nv	ttd
LC1_2_1	**20**	**2704.57**	**20**	**2704.57**	**20**	**2704.57**
LC1_2_2	**19**	**2764.56**	**19**	**2764.56**	**19**	**2764.56**
LC1_2_3	**17**	**3127.78***	17	3128.61	17	3128.61
LC1_2_4	**17**	**2693.41**	**17**	**2693.41**	**17**	**2693.41**
LC1_2_5	**20**	**2702.05**	**20**	**2702.05**	**20**	**2702.05**
LC1_2_6	**20**	**2701.04**	**20**	**2701.04**	**20**	**2701.04**
LC1_2_7	**20**	**2701.04**	**20**	**2701.04**	**20**	**2701.04**
LC1_2_8	**20**	**2689.83**	**20**	**2689.83**	**20**	**2689.83**
LC1_2_9	**18**	**2724.24**	**18**	**2724.24**	**18**	**2724.24**
LC1_2_10	**17**	**2942.13***	17	2943.49	17	2943.49
LC2_2_1	**6**	**1931.44**	**6**	**1931.44**	**6**	**1931.44**
LC2_2_2	**6**	**1881.40**	**6**	**1881.40**	**6**	**1881.40**
LC2_2_3	**6**	**1844.33**	**6**	**1844.33**	**6**	**1844.33**
LC2_2_4	6	1767.82	**6**	**1767.12**	**6**	**1767.12**
LC2_2_5	**6**	**1891.21**	**6**	**1891.21**	**6**	**1891.21**
LC2_2_6	**6**	**1857.78**	**6**	**1857.78**	**6**	**1857.78**
LC2_2_7	**6**	**1850.13**	**6**	**1850.13**	**6**	**1850.13**
LC2_2_8	**6**	**1824.34**	**6**	**1824.34**	**6**	**1824.34**
LC2_2_9	**6**	**1854.21**	**6**	**1854.21**	**6**	**1854.21**
LC2_2_10	**6**	**1817.45**	**6**	**1817.45**	**6**	**1817.45**
LR1_2_1	**20**	**4819.12**	**20**	**4819.12**	**20**	**4819.12**
LR1_2_2	**17**	**4621.21**	**17**	**4621.21**	**17**	**4621.21**
LR1_2_3	**15**	**3612.64**	**15**	**3612.64**	**15**	**3612.64**
LR1_2_4	**10**	**3031.20***	10	3037.38	10	3037.38
LR1_2_5	**16**	**4760.18**	**16**	**4760.18**	**16**	**4760.18**
LR1_2_6	**14**	**4175.16**	14	4178.24	**14**	**4175.16**
LR1_2_7	**12**	**3543.56***	12	3550.61	12	3550.61
LR1_2_8	9	2791.67	**9**	**2784.53**	**9**	**2784.53**
LR1_2_9	**14**	**4343.86***	14	4354.66	14	4354.66
LR1_2_10	**11**	**3695.84***	11	3714.16	11	3714.16
LR2_2_1	**5**	**4073.10**	**5**	**4073.10**	**5**	**4073.10**
LR2_2_2	4	3796.81	**4**	**3796.00**	**4**	**3796.00**
LR2_2_3	**4**	**3098.36**	**4**	**3098.36**	**4**	**3098.36**
LR2_2_4	3	2500.04	**3**	**2486.14**	**3**	**2486.14**
LR2_2_5	**4**	**3438.39**	**4**	**3438.39**	**4**	**3438.39**
LR2_2_6	**4**	**3201.54**	**4**	**3201.54**	**4**	**3201.54**
LR2_2_7	3	3152.52	**3**	**3135.05**	**3**	**3135.05**
LR2_2_8	2	2582.35	**2**	**2555.40**	**2**	**2555.40**
LR2_2_9	3	4054.50	**3**	**3930.49**	**3**	**3930.49**
LR2_2_10	**3**	**3286.95***	3	3344.08	3	3323.37
LRC1_2_1	**19**	**3606.06**	**19**	**3606.06**	**19**	**3606.06**
LRC1_2_2	15	3681.07	15	3674.80	**15**	**3671.02**
LRC1_2_3	**13**	**3154.92***	13	3178.17	13	3161.75
LRC1_2_4	**10**	**2631.82**	**10**	**2631.82**	**10**	**2631.82**
LRC1_2_5	**16**	**3715.81**	**16**	**3715.81**	**16**	**3715.81**
LRC1_2_6	17	3368.66	17	3368.66	**16**	**3572.16**
LRC1_2_7	14	3738.47	**14**	**3668.39**	**14**	**3668.39**
LRC1_2_8	13	3167.23	13	3174.55	**13**	**3146.70**
LRC1_2_9	13	3303.18	13	3226.72	**13**	**3157.34**
LRC1_2_10	12	2951.90	**12**	**2951.29**	**12**	**2951.29**
LRC2_2_1	6	3632.37	6	3605.40	**6**	**3595.18**
LRC2_2_2	**5**	**3182.62***	5	3327.18	5	3327.18
LRC2_2_3	**4**	**2914.82***	4	2938.28	4	2938.28
LRC2_2_4	3	3038.16	**3**	**2887.97**	**3**	**2887.97**
LRC2_2_5	5	2777.23	**5**	**2776.93**	**5**	**2776.93**
LRC2_2_6	**5**	**2707.96**	**5**	**2707.96**	**5**	**2707.96**
LRC2_2_7	4	3067.91	4	3056.09	**4**	**3044.40**
LRC2_2_8	4	2401.17	**4**	**2399.95**	**4**	**2399.95**
LRC2_2_9	4	2208.72	**4**	**2208.49**	**4**	**2208.49**
LRC2_2_10	3	2600.41	**3**	**2550.56**	**3**	**2550.56**
Average		**3005.62**		**3000.85**		**2998.08**

cluster" individual clusters may contain pickup as well as delivery points, while only sites of one sort can occur in an individual cluster of variant "Pure clusters". For each instance size, the same number of instances belongs to each distribution variant.

- Loading spaces and boxes are generated similarly to Gendreau et al. (2006). The dimensions of the uniform loading spaces of the vehicles are chosen as $L = 60$, $W = 25$ and $H = 30$ length units. The lengths l_{ik}, widths w_{ik} and heights h_{ik} of the boxes are drawn randomly from the intervals $[0.2{\cdot}L, 0.6{\cdot}L]$, $[0.2{\cdot}W, 0.6{\cdot}W]$ and $[0.2{\cdot}H, 0.6{\cdot} H]$, respectively ($i = 1,\ldots,n$, $k = 1,\ldots,m_i$). A box is characterized as fragile with the probability 0.25. The percentage a for the minimal supporting area was specified as 0.75.
- The weight capacity of the loading spaces was set to 45,000 weight units (being also the value of volume). The proportion of weight and volume is chosen three to one for the boxes of one third of the requests of an instance while for the boxes of the residual requests identical values are chosen for weights and

Table 10
Results (travel distances) for different variants of 3L-PDP.

Instance	1D-test	Variant 1A		Variant 1B		Variant 2		Variant 5	
	ttd	ttd	Gap (percent)	ttd	Gap (percent)	ttd	Gap (percent)	ttd	Gap (percent)
50_RAND_2_1	1362.26	1455.90	6.87	1465.44	7.57	1630.49	19.69	1732.67	27.19
50_RAND_2_2	1181.42	1320.96	11.81	1325.90	12.23	1519.02	28.58	1583.34	34.02
50_RAND_2_3	1234.98	1329.43	7.65	1343.59	8.79	1568.65	27.02	1650.54	33.65
50_RAND_2_4	1246.39	1368.32	9.78	1374.41	10.27	1536.86	23.31	1602.59	28.58
50_RAND_2_5	1276.39	1338.52	4.87	1346.96	5.53	1540.24	20.67	1595.81	25.03
50_CLUS_2_1	888.14	993.52	11.87	986.23	11.04	1052.67	18.53	1123.83	26.54
50_CLUS_2_2	852.82	929.07	8.94	926.97	8.69	1035.76	21.45	1111.18	30.29
50_CLUS_2_3	921.30	992.28	7.70	1005.00	9.08	1099.69	19.36	1150.02	24.83
50_CLUS_2_4	1031.28	1111.93	7.82	1122.61	8.86	1222.88	18.58	1275.35	23.67
50_CLUS_2_5	1132.02	1238.50	9.41	1231.13	8.76	1308.56	15.60	1381.87	22.07
50_CPCD_2_1	1102.82	1273.17	15.45	1277.44	15.83	1307.80	18.59	1378.21	24.97
50_CPCD_2_2	1039.63	1165.22	12.08	1182.16	13.71	1245.08	19.76	1257.93	21.00
50_CPCD_2_3	996.64	1118.85	12.26	1127.60	13.14	1193.27	19.73	1235.76	23.99
50_CPCD_2_4	1128.00	1264.06	12.06	1266.33	12.26	1305.44	15.73	1334.87	18.34
50_CPCD_2_5	1237.93	1378.02	11.32	1394.96	12.68	1426.37	15.22	1457.98	17.78
50_RAND_3_1	1359.03	1468.96	8.09	1468.08	8.02	1615.05	18.84	1729.64	27.27
50_RAND_3_2	1184.75	1347.39	13.73	1321.31	11.53	1469.79	24.06	1574.71	32.91
50_RAND_3_3	1251.30	1339.55	7.05	1381.46	10.40	1563.98	24.99	1657.33	32.45
50_RAND_3_4	1266.16	1340.17	5.85	1351.32	6.73	1543.46	21.90	1569.01	23.92
50_RAND_3_5	1281.81	1350.06	5.32	1367.22	6.66	1542.28	20.32	1589.65	24.02
50_CLUS_3_1	889.57	975.26	9.63	977.07	9.84	1007.59	13.27	1052.79	18.35
50_CLUS_3_2	854.41	912.67	6.82	927.89	8.60	1027.42	20.25	1104.59	29.28
50_CLUS_3_3	926.75	992.13	7.05	1014.05	9.42	1083.43	16.91	1126.70	21.58
50_CLUS_3_4	1026.55	1106.48	7.79	1110.01	8.13	1192.62	16.18	1251.52	21.92
50_CLUS_3_5	1137.96	1235.78	8.60	1242.11	9.15	1281.10	12.58	1324.79	16.42
50_CPCD_3_1	1097.01	1279.04	16.59	1311.43	19.55	1331.47	21.37	1352.20	23.26
50_CPCD_3_2	1048.17	1212.34	15.66	1216.30	16.04	1233.94	17.72	1248.24	19.09
50_CPCD_3_3	998.97	1163.52	16.47	1146.02	14.72	1217.32	21.86	1243.72	24.50
50_CPCD_3_4	1135.69	1277.65	12.50	1277.68	12.50	1316.08	15.88	1319.36	16.17
50_CPCD_3_5	1237.80	1406.17	13.60	1400.27	13.13	1434.46	15.89	1456.91	17.70
75_RAND_2_1	1701.32	1869.07	9.86	1837.70	8.02	2038.57	19.82	2133.26	25.39
75_RAND_2_2	1562.31	1759.52	12.62	1770.07	13.30	2044.91	30.89	2135.29	36.68
75_RAND_2_3	1653.16	1838.57	11.22	1853.53	12.12	2100.18	27.04	2184.47	32.14
75_CLUS_2_1	1180.71	1309.51	10.91	1312.78	11.19	1392.76	17.96	1468.07	24.34
75_CLUS_2_2	1136.71	1269.25	11.66	1273.33	12.02	1374.76	20.94	1422.51	25.14
75_CLUS_2_3	1163.14	1325.77	13.98	1333.76	14.67	1435.62	23.43	1500.43	29.00
75_CPCD_2_1	1803.87	2033.94	12.75	2048.82	13.58	2187.43	21.26	2245.93	24.51
75_CPCD_2_2	1792.68	2050.49	14.38	2045.19	14.09	2198.82	22.66	2223.93	24.06
75_CPCD_2_3	1879.48	2103.61	11.92	2128.39	13.24	2236.83	19.01	2290.03	21.84
75_RAND_3_1	1698.67	1881.72	10.78	1892.65	11.42	2048.80	20.61	2155.38	26.89
75_RAND_3_2	1562.97	1721.06	10.11	1728.66	10.60	1969.33	26.00	2078.85	33.01
75_RAND_3_3	1654.16	1772.72	7.17	1808.46	9.33	2050.93	23.99	2129.98	28.77
75_CLUS_3_1	1178.62	1326.51	12.55	1334.40	13.22	1420.35	20.51	1455.27	23.47
75_CLUS_3_2	1138.91	1258.87	10.53	1299.42	14.09	1402.37	23.13	1433.76	25.89
75_CLUS_3_3	1173.96	1338.52	14.02	1305.07	11.17	1424.94	21.38	1478.68	25.96
75_CPCD_3_1	1815.60	2103.01	15.83	2100.26	15.68	2190.94	20.67	2233.76	23.03
75_CPCD_3_2	1810.81	2088.58	15.34	2089.01	15.36	2207.67	21.92	2217.93	22.48
75_CPCD_3_3	1883.97	2154.68	14.37	2190.03	16.25	2247.28	19.28	2264.96	20.22
100_RAND_2_1	3086.28	3518.97	14.02	3530.30	14.39	4028.80	30.54	4087.70	32.45
100_CLUS_2_1	3176.76	3681.39	15.89	3695.87	16.34	3998.53	25.87	4195.66	32.07
100_CPCD_2_1	3541.89	4200.16	18.59	4161.98	17.51	4315.08	21.83	4342.17	22.59
100_RAND_3_1	3110.14	3534.29	13.64	3532.25	13.57	4006.17	28.81	4079.84	31.18
100_CLUS_3_1	3206.88	3714.74	15.84	3768.41	17.51	4022.36	25.43	4169.56	30.02
100_CPCD_3_1	3579.25	4244.14	18.58	4281.03	19.61	4310.86	20.44	4245.06	18.60
Average gap			**11.50**		**12.06**		**21.06**		**25.38**

volumes. Hereby the weight constraint (C5) does not become redundant.

- The maximal number of admitted vehicles v_{max} per instance is determined such that the algorithm can relatively easily find a feasible solution for problem variant 5. Since we expect shorter travel distances for the other problem variants, it should be possible to find feasible solutions for the other problem variants which respect the determined value of v_{max}, too.

The 54 new 3L-PDP instances are overviewed in Table 4; the figures in columns 2–8 are instance numbers. The instances are offered at the website http://www.mansci.ovgu.de/Forschung /Materialien.html.

5.2. Parameter setting

The parameter setting for the experiments is specified in Tables 5 and 6. The same parameterization of the routing procedures is used for all problem variants. All parameter values were determined based on limited computational experiments using a trial and error strategy.

In Table 7 the maximum run time per instance and single run is shown. The computing time depends on the number of requests and the average box number per request. Lower time limits are specified if the new PDP instances are tested by 1D-LNS, i.e. as 1D-PDP instances.

Table 11
Results (reloading efforts) for different variants of 3L-PDP.

Instance	Cargo Weight	Variant 1A Reloading effort		Variant 1B Reloading effort		Variant 2 Reloading effort	
		Absolute	in percent	Absolute	in percent	Absolute	in percent
50_RAND_2_1	610,544	505,726	82.83	459,168	75.21	66,021	10.81
50_RAND_2_2	578,322	612,823	105.97	553,046	95.63	97,379	16.84
50_RAND_2_3	530,415	655,020	123.49	599,036	112.94	142,204	26.81
50_RAND_2_4	652,932	590,308	90.41	487,152	74.61	35,340	5.41
50_RAND_2_5	698,040	508,646	72.87	465,401	66.67	93,769	13.43
50_CLUS_2_1	610,544	488,239	79.97	501,062	82.07	82,880	13.57
50_CLUS_2_2	578,322	553,771	95.75	514,097	88.89	248,195	42.92
50_CLUS_2_3	530,415	575,112	108.43	497,441	93.78	120,055	22.63
50_CLUS_2_4	652,932	586,638	89.85	573,017	87.76	97,995	15.01
50_CLUS_2_5	698,040	472,835	67.74	370,133	53.02	109,537	15.69
50_CPCD_2_1	610,544	482,635	79.05	453,920	74.35	45,237	7.41
50_CPCD_2_2	578,322	521,266	90.13	477,350	82.54	43,122	7.46
50_CPCD_2_3	530,415	449,272	84.70	505,457	95.29	39,877	7.52
50_CPCD_2_4	652,932	331,524	50.77	347,931	53.29	40,816	6.25
50_CPCD_2_5	698,040	451,510	64.68	407,285	58.35	28,235	4.04
50_RAND_3_1	611,295	460,657	75.36	466,731	76.35	80,832	13.22
50_RAND_3_2	579,037	578,259	99.87	457,797	79.06	107,977	18.65
50_RAND_3_3	531,236	636,004	119.72	599,437	112.84	101,077	19.03
50_RAND_3_4	654,049	577,658	88.32	507,945	77.66	47,585	7.28
50_RAND_3_5	699,080	495,528	70.88	498,884	71.36	54,599	7.81
50_CLUS_3_1	611,295	504,901	82.60	473,175	77.41	114,401	18.71
50_CLUS_3_2	579,037	468,392	80.89	449,459	77.62	93,617	16.17
50_CLUS_3_3	531,236	600,787	113.09	509,027	95.82	101,148	19.04
50_CLUS_3_4	654,049	591,051	90.37	509,588	77.91	74,302	11.36
50_CLUS_3_5	699,080	464,924	66.51	422,781	60.48	75,528	10.80
50_CPCD_3_1	611,295	424,166	69.39	435,437	71.23	35,762	5.85
50_CPCD_3_2	579,037	494,572	85.41	475,147	82.06	45,572	7.87
50_CPCD_3_3	531,236	443,460	83.48	436,738	82.21	50,518	9.51
50_CPCD_3_4	654,049	405,999	62.07	339,285	51.87	60,050	9.18
50_CPCD_3_5	699,080	471,397	67.43	372,119	53.23	31,736	4.54
75_RAND_2_1	772,435	858,913	111.20	744,474	96.38	127,389	16.49
75_RAND_2_2	780,361	906,935	116.22	795,663	101.96	124,954	16.01
75_RAND_2_3	808,203	910,351	112.64	810,807	100.32	116,672	14.44
75_CLUS_2_1	772,435	912,196	118.09	734,429	95.08	96,284	12.46
75_CLUS_2_2	780,361	781,937	100.20	678,881	87.00	118,423	15.18
75_CLUS_2_3	808,203	903,054	111.74	839,403	103.86	152,376	18.85
75_CPCD_2_1	772,435	766,092	99.18	721,262	93.38	106,943	13.84
75_CPCD_2_2	780,361	742,619	95.16	704,138	90.23	88,685	11.36
75_CPCD_2_3	808,203	712,251	88.13	701,047	86.74	33,692	4.17
75_RAND_3_1	774,140	885,903	114.44	719,354	92.92	96,865	12.51
75_RAND_3_2	782,381	928,207	118.64	786,472	100.52	128,936	16.48
75_RAND_3_3	810,106	913,731	112.79	787,292	97.18	85,975	10.61
75_CLUS_3_1	774,140	828,317	107.00	710,133	91.73	143,045	18.48
75_CLUS_3_2	782,381	813,616	103.99	710,365	90.80	181,135	23.15
75_CLUS_3_3	810,106	970,844	119.84	872,397	107.69	97,142	11.99
75_CPCD_3_1	774,140	855,853	110.56	709,599	91.66	104,236	13.46
75_CPCD_3_2	782,381	725,508	92.73	673,624	86.10	155,560	19.88
75_CPCD_3_3	810,106	751,590	92.78	667,262	82.37	47,472	5.86
100_RAND_2_1	1,072,407	1,202,262	112.11	1,071,590	99.92	121,210	11.30
100_CLUS_2_1	1,072,407	1,228,476	114.55	1,034,802	96.49	183,258	17.09
100_CPCD_2_1	1,072,407	1,040,541	97.03	1,053,133	98.20	183,908	17.15
100_RAND_3_1	1,074,809	1,401,150	130.36	1,161,062	108.02	98,048	9.12
100_CLUS_3_1	1,074,809	1,245,867	115.92	1,085,592	101.00	126,403	11.76
100_CPCD_3_1	1,074,809	1,093,020	101.69	991,355	92.24	84,524	7.86
Average			**95.17**		**85.80**		**13.41**

For the 1D-LNS test by means of the instances by Li and Lim (2001) with 50 (respectively 100) requests, we allow a computing time of two (respectively five) minutes.

5.3. Computational results for the one-dimensional PDP(TW)

The PDP instances by Li and Lim (2001) are actually PDPTW instances, i.e. the requests have time windows. To perform a comparison with other algorithms using the Li and Lim instances, we had to extend our hybrid algorithm to make it capable of taking time windows into account. Furthermore, most of the existing PDPTW solution procedures do minimize primarily the number of used vehicles (or routes) and the total travel distance is only the second objective criterion. We did adapt our hybrid algorithm also in this regard by a two-phase approach. In the first phase the number of routes is minimized while the travel distance is minimized in the second one without sacrificing the reached number of vehicles.

We tested 1D-LNS using the Li and Lim instances with 50 and 100 requests (100 and 200 customers, respectively) and compared the results of 1D-LNS with those of Ropke and Pisinger (2006) and the best known solutions. These were taken from the website Sintef (2015) and from Koning (2011).

Tables 8 and 9 present the best solutions achieved by 1D-LNS and by Ropke and Pisinger (2006) as well as the best known solutions as indicated in the above sources (*nv* is the number of used vehicles, *ttd* is the total travel distance). Best values are in bold;

Table 12
Tradeoff between total travel distance and reloading effort.

3L-PDP variant	Total travel distance average (percent)	Reloading effort average (percent)
1A	111.50	95.17
1B	112.06	85.80
2	121.06	13.14
5	125.38	0.00

new best solutions are marked by an asterisk (*). For 55 of 56 Li-Lim-100 instances 1D-LNS finds the best known solution and reaches exactly the same results as ALNS by Ropke and Pisinger (2006) for all 56 instances. For the Li-Lim-200 instances 1D-LNS performs slightly worse than ALNS. Both algorithms achieve the minimal known number of vehicles for 59 of 60 instances. The average gap of the total travel distance amounts to 0.25 percent for 1D-LNS and to 0.09 percent for ALNS. Please note that in Table 9 the average value is calculated over 59 instances; LRC_1_2_6 is omitted due to the different number of vehicles in the best solutions.

For a single instance, the gap is determined as (*ttd-best* – *ttd-best-known*)/*ttd-best-known* (in percent) where *ttd-best* is the best reached travel distance (over ten runs) and *ttd-best-known* is the best known travel distance for the given instance. Moreover, 1D-LNS achieves new best solutions for ten instances. Here we have to make the restrictive remark that we only claim to achieve

Table 13
Computing times to find the best solution for different variants of 3L-PDP.

Instance	Variant 1A		Variant 1B		Variant 2		Variant 5	
	Seconds	Percent	Seconds	Percent	Seconds	Percent	Seconds	Percent
50_RAND_2_1	107.17	89.31	91.60	76.33	71.64	59.70	58.39	48.66
50_RAND_2_2	87.36	72.80	102.27	85.23	92.42	77.02	81.87	68.23
50_RAND_2_3	101.05	84.21	100.60	83.83	62.78	52.32	64.71	53.93
50_RAND_2_4	95.84	79.87	106.10	88.42	66.86	55.72	54.53	45.44
50_RAND_2_5	81.23	67.69	110.79	92.33	52.84	44.03	31.68	26.40
50_CLUS_2_1	94.86	79.05	91.60	76.33	71.32	59.43	41.33	34.44
50_CLUS_2_2	110.40	92.00	94.37	78.64	95.83	79.86	96.57	80.48
50_CLUS_2_3	88.95	74.13	102.35	85.29	79.60	66.33	70.88	59.07
50_CLUS_2_4	91.09	75.91	113.16	94.30	72.94	60.78	73.71	61.43
50_CLUS_2_5	83.58	69.65	102.00	85.00	58.55	48.79	66.38	55.32
50_CPCD_2_1	100.36	83.63	111.38	92.82	91.76	76.47	88.20	73.50
50_CPCD_2_2	102.80	85.67	95.55	79.63	101.48	84.57	101.20	84.33
50_CPCD_2_3	113.46	94.55	97.20	81.00	89.91	74.93	86.61	72.18
50_CPCD_2_4	94.99	79.16	93.93	78.28	103.31	86.09	88.51	73.76
50_CPCD_2_5	71.25	59.38	89.83	74.86	103.10	85.92	86.82	72.35
50_RAND_3_1	239.55	79.85	263.57	87.86	190.23	63.41	186.90	62.30
50_RAND_3_2	228.76	76.25	247.94	82.65	257.26	85.75	265.20	88.40
50_RAND_3_3	247.53	82.51	276.61	92.20	244.00	81.33	203.02	67.67
50_RAND_3_4	230.28	76.76	268.44	89.48	245.11	81.70	200.30	66.77
50_RAND_3_5	215.01	71.67	240.22	80.07	189.05	63.02	195.87	65.29
50_CLUS_3_1	261.56	87.19	229.32	76.44	200.31	66.77	205.10	68.37
50_CLUS_3_2	261.52	87.17	254.47	84.82	231.87	77.29	190.09	63.36
50_CLUS_3_3	229.92	76.64	257.62	85.87	231.30	77.10	237.31	79.10
50_CLUS_3_4	268.27	89.42	256.52	85.51	197.22	65.74	243.49	81.16
50_CLUS_3_5	216.10	72.03	221.72	73.91	183.57	61.19	204.54	68.18
50_CPCD_3_1	287.19	95.73	281.54	93.85	248.07	82.69	200.12	66.71
50_CPCD_3_2	263.59	87.86	261.27	87.09	245.65	81.88	206.89	68.96
50_CPCD_3_3	265.75	88.58	259.45	86.48	250.86	83.62	265.12	88.37
50_CPCD_3_4	247.67	82.56	268.70	89.57	184.25	61.42	255.28	85.09
50_CPCD_3_5	257.15	85.72	267.74	89.25	229.38	76.46	258.38	86.13
75_RAND_2_1	210.26	87.61	186.73	77.80	189.02	78.76	159.63	66.51
75_RAND_2_2	201.76	84.07	205.09	85.45	205.11	85.46	148.71	61.96
75_RAND_2_3	190.70	79.46	203.51	84.80	182.27	75.95	171.46	71.44
75_CLUS_2_1	201.62	84.01	177.04	73.77	178.44	74.35	157.94	65.81
75_CLUS_2_2	196.87	82.03	207.99	86.66	190.69	79.45	150.77	62.82
75_CLUS_2_3	181.51	75.63	197.47	82.28	181.04	75.43	201.00	83.75
75_CPCD_2_1	224.35	93.48	186.97	77.90	208.02	86.68	194.12	80.88
75_CPCD_2_2	166.59	69.41	174.99	72.91	200.35	83.48	221.48	92.28
75_CPCD_2_3	202.99	84.58	183.48	76.45	204.10	85.04	172.84	72.02
75_RAND_3_1	493.66	82.28	468.80	78.13	491.12	81.85	438.72	73.12
75_RAND_3_2	523.75	87.29	495.64	82.61	456.54	76.09	455.96	75.99
75_RAND_3_3	467.60	77.93	458.48	76.41	468.25	78.04	491.88	81.98
75_CLUS_3_1	550.75	91.79	540.45	90.08	531.09	88.52	521.55	86.93
75_CLUS_3_2	481.92	80.32	552.59	92.10	561.17	93.53	434.24	72.37
75_CLUS_3_3	475.03	79.17	453.02	75.50	496.14	82.69	530.26	88.38
75_CPCD_3_1	510.26	85.04	547.66	91.28	491.39	81.90	508.21	84.70
75_CPCD_3_2	540.20	90.03	505.86	84.31	492.51	82.09	498.55	83.09
75_CPCD_3_3	511.63	85.27	559.80	93.30	488.83	81.47	482.27	80.38
100_RAND_2_1	393.55	81.99	340.08	70.85	408.82	85.17	410.75	85.57
100_CLUS_2_1	398.93	83.11	420.58	87.62	442.69	92.23	406.45	84.68
100_CPCD_2_1	407.49	84.89	408.01	85.00	363.87	75.81	402.10	83.77
100_RAND_3_1	1028.53	85.71	1049.58	87.47	998.42	83.20	884.72	73.73
100_CLUS_3_1	861.88	71.82	991.09	82.59	916.64	76.39	936.48	78.04
100_CPCD_3_1	1073.93	89.49	1130.31	94.19	1025.17	85.43	1055.99	88.00
Average		**81.91**		**83.83**		**75.75**		**72.10**

Table 14
Maximal loaded volumes for different variants of 3L-PDP.

Instance	Variant 1A		Variant 1B		Variant 2		Variant 5	
	Volume	Percent	Volume	Percent	Volume	Percent	Volume	Percent
50_RAND_2_1	28,847	64.10	28,718	63.82	28,040	62.31	23,641	52.53
50_RAND_2_2	27,836	61.86	29,092	64.65	26,561	59.02	26,039	57.86
50_RAND_2_3	27,914	62.03	27,099	60.22	26,405	58.68	24,215	53.81
50_RAND_2_4	27,993	62.21	26,471	58.82	24,781	55.07	24,963	55.47
50_RAND_2_5	29,284	65.07	29,022	64.49	27,368	60.82	26,970	59.93
50_CLUS_2_1	29,523	65.61	28,784	63.96	28,208	62.68	28,807	64.01
50_CLUS_2_2	28,009	62.24	27,312	60.69	26,602	59.12	24,562	54.58
50_CLUS_2_3	27,642	61.43	28,297	62.88	27,534	61.19	25,204	56.01
50_CLUS_2_4	27,909	62.02	28,071	62.38	26,514	58.92	24,727	54.95
50_CLUS_2_5	30,359	67.47	28,844	64.10	29,714	66.03	26,622	59.16
50_CPCD_2_1	29,619	65.82	30,377	67.50	29,916	66.48	27,557	61.24
50_CPCD_2_2	29,897	66.44	29,669	65.93	29,348	65.22	29,182	64.85
50_CPCD_2_3	29,136	64.75	29,211	64.91	27,837	61.86	28,377	63.06
50_CPCD_2_4	28,894	64.21	28,103	62.45	27,449	61.00	24,984	55.52
50_CPCD_2_5	30,546	67.88	31,818	70.71	30,406	67.57	30,363	67.47
50_RAND_3_1	28,824	64.05	26,740	59.42	25,971	57.71	28,857	64.13
50_RAND_3_2	26,682	59.29	28,484	63.30	27,371	60.82	26,065	57.92
50_RAND_3_3	29,712	66.03	27,022	60.05	27,154	60.34	27,073	60.16
50_RAND_3_4	25,861	57.47	25,066	55.70	25,833	57.41	25,801	57.33
50_RAND_3_5	26,536	58.97	27,418	60.93	27,590	61.31	27,750	61.67
50_CLUS_3_1	28,120	62.49	28,899	64.22	27,561	61.25	28,226	62.72
50_CLUS_3_2	28,435	63.19	28,505	63.34	25,889	57.53	25,450	56.56
50_CLUS_3_3	27,006	60.01	27,798	61.77	24,391	54.20	24,379	54.18
50_CLUS_3_4	24,989	55.53	26,459	58.80	24,891	55.31	24,882	55.29
50_CLUS_3_5	31,230	69.40	31,268	69.48	32,116	71.37	29,504	65.56
50_CPCD_3_1	29,705	66.01	29,090	64.64	28,875	64.17	28,858	64.13
50_CPCD_3_2	29,215	64.92	28,621	63.60	29,348	65.22	29,225	64.94
50_CPCD_3_3	27,826	61.83	29,134	64.74	28,336	62.97	27,813	61.81
50_CPCD_3_4	28,206	62.68	28,167	62.59	29,009	64.46	27,285	60.63
50_CPCD_3_5	30,025	66.72	30,645	68.10	29,885	66.41	30,062	66.81
75_RAND_2_1	27,802	61.78	28,468	63.26	28,373	63.05	26,987	59.97
75_RAND_2_2	28,655	63.68	29,201	64.89	28,510	63.36	27,962	62.14
75_RAND_2_3	29,885	66.41	28,616	63.59	26,875	59.72	27,749	61.66
75_CLUS_2_1	29,810	66.24	28,451	63.22	29,056	64.57	27,768	61.71
75_CLUS_2_2	30,501	67.78	31,098	69.11	30,494	67.76	30,366	67.48
75_CLUS_2_3	29,119	64.71	28,473	63.27	27,822	61.83	27,214	60.48
75_CPCD_2_1	30,478	67.73	30,314	67.36	27,860	61.91	27,813	61.81
75_CPCD_2_2	29,922	66.49	29,280	65.07	29,093	64.65	27,893	61.98
75_CPCD_2_3	29,905	66.46	29,843	66.32	30,440	67.64	29,844	66.32
75_RAND_3_1	28,451	63.22	27,095	60.21	26,547	58.99	25,637	56.97
75_RAND_3_2	27,907	62.01	27,843	61.87	25,142	55.87	27,800	61.78
75_RAND_3_3	26,623	59.16	28,105	62.46	27,769	61.71	27,742	61.65
75_CLUS_3_1	27,775	61.72	27,125	60.28	27,806	61.79	27,882	61.96
75_CLUS_3_2	27,959	62.13	27,946	62.10	28,113	62.47	25,096	55.77
75_CLUS_3_3	27,311	60.69	27,408	60.91	27,188	60.42	25,228	56.06
75_CPCD_3_1	28,533	63.41	28,481	63.29	27,818	61.82	28,537	63.41
75_CPCD_3_2	28,731	63.85	28,046	62.32	28,676	63.72	28,115	62.48
75_CPCD_3_3	28,779	63.95	30,682	68.18	28,508	63.35	27,763	61.70
100_RAND_2_1	29,108	64.68	28,516	63.37	29,149	64.78	27,690	61.53
100_CLUS_2_1	30,501	67.78	31,020	68.93	29,120	64.71	27,842	61.87
100_CPCD_2_1	30,387	67.53	29,773	66.16	29,156	64.79	30,317	67.37
100_RAND_3_1	27,881	61.96	28,541	63.42	27,822	61.83	29,131	64.74
100_CLUS_3_1	29,157	64.79	28,604	63.56	27,881	61.96	28,569	63.49
100_CPCD_3_1	30,420	67.60	27,941	62.09	27,889	61.98	27,811	61.80
Average		**63.84**		**63.58**		**62.06**		**60.75**

(old or new) best solutions with regard to the above mentioned references.

Our LNS implementation reaches nearly the solution quality of the best available methods for the PDPTW. A high solution quality was achieved although the LNS has been rather simplified compared to the original procedure by Ropke and Pisinger (2006). As limitation has to be noted that the results shown by Ropke and Pisinger (2006) are achieved with comparable computing times of average of 49 seconds (Li-Lim-100) and 305 seconds (Li-Lim-200) but with a significantly weaker hardware (Intel Pentium IV with 1.5 gigahertz). In contrast, an Intel Core i5-2500 K with 4.0 gigahertz was used for 1D-LNS, the computing times were 120 and 300 seconds for the Li-Lim instances with 50 and 100 requests.

5.4. Computational results for the 3L-PDP

The detailed results for the 3L-PDP instances regarding total travel distance (*ttd*) are presented in Table 10. In the leftmost column the instance names are listed. The next column shows the total travel distances for the 1D test for which only the weight constraint (C5), and the routing constraints (C9) and (C10) are considered. In the following eight columns the total travel distances and the gaps are indicated for the 3L-PDP variants 1A, 1B, 2 and 5 (see Table 2). All presented total travel distances are mean values over five runs. The corresponding gaps are calculated as (*ttd - ttd-1D*) / *ttd-1D* * 100 (percent). In the last line of Table 10 the gap values of the 3L-PDP variants are averaged over the 54 instances.

Summarizing the results, we can state that the travel distances increase significantly if the 3L-PDP instances are solved instead of the corresponding 1D-PDP instances. For the problem variants 1A and 1B, the total travel distances grow on average by 11.50 percent and by 12.06 percent, respectively, compared to the 1D case. For the problem variants 2 and 5, the mean increase is even higher and amounts to 21.06 percent and 25.38 percent, respectively.

In Table 11 the reloading effort for the 3L-PDP variants 1A, 1B and 2 is indicated. Since the reloading effort is zero for problem variant 5, this variant does not occur in Table 11 (see Table 2).

The reloading effort needed for a 3L-PDP instance is given as the weight of all boxes that are reloaded. If a box is reloaded, say, three times the weight of the box is counted three times. Thus it may occur that the reloading effort exceeds the total weight of boxes.

Table 11 is organized as follows. The first column includes the instance names and the second column shows the total weight of all requests per instance (cargo weight). In the following six columns the reloading efforts for the relevant 3L-PDP variants are given as absolute values (in weight units) and as percentages of the cargo weight. The results per instance are, again, averaged over five runs. In the last line of Table 11 the percentaged reloading efforts are averaged over the 54 instances.

The reloading effort of problem variant 2, caused by the missing reloading ban, is only moderate and amounts to 13.41 percent of the cargo weight on average. For problem variants 1A and 1B the mean reloading effort is much higher (95.17 percent and 85.80 percent, respectively), since not only the reloading ban is missing but the RS constraint (C2) for delivery points is not required, either. The difference of 9.37 percentage points between the subvariants of problem variant 1 is also plausible, since in subvariant 1B the reloading effort is reduced by a better loading order of the affected requests.

Table 12 summarizes the results regarding total travel distance and reloading effort. For each 3L-PDP variant, the total travel distance is here given as percentage of the travel distance in the 1D-case while the reloading efforts are again indicated as percentages of the cargo weight. All presented values are averaged over the five runs per instance and over the 54 3L-PDP instances.

The results clearly show that a 3D modeling of pickup and delivery problems leads to a significant increase of the travel distances in the 3D case and is, therefore, generally more realistic. The indicated figures for the 3L-PDP variants correspond very well with the expected differences between those variants regarding travel distances and reloading effort as shown in Table 2. The main result is the clear tradeoff between travel distances and reloading effort indicating that a saving of travel distance has to be "paid" with an additional portion of reloading effort.

In Table 13 the average computing times to find the best solution are shown. The times are given as absolute values in seconds and as percentages of the maximum allowed computing time per instance (see Table 7). All values are averaged over five runs. The results in Table 13 show that there is still potential for improvement, i.e. the algorithms should be able to find better solutions within higher computing times.

In Table 14 the maximal volume of contained boxes is shown which the vehicles are reaching within their tours. The maximal loaded volumes are averaged over the routes and are given as absolute values and as percentages of the vehicles loading space volume (45,000). Again all values are averaged over five runs. For all problem variants an average maximum space utilization greater 0.6 is reached, hence a quite good utilization of the available loading space.

6. Conclusions and future work

In this paper, the vehicle routing problem with pickup and delivery (PDP) has been extended to an integrated vehicle routing and loading problem with 3D rectangular items to be transported in homogeneous vehicles with a rectangular 3D loading space (3L-PDP). In the problem formulation, we focused on the question under which conditions any reloading effort, i.e. any movement of boxes *after* loading and *before* unloading, can be avoided. It turned out that the request sequence constraints for pickup and delivery points are not sufficient. Instead, we require either a new packing constraint, termed reloading ban, or a new routing constraint, called independent partial routes condition, to exclude any reloading effort. Eventually, a spectrum of five 3L-PDP variants was introduced that allow for different portions of reloading effort and reciprocal savings of travel distance.

A hybrid algorithm was then proposed to tackle three of the five 3L-PDP variants. It is composed of the large neighborhood search algorithm by Ropke and Pisinger (2006) for the 1D-PDP and the tree search algorithm for packing boxes by Bortfeldt (2012). We tested the algorithm with the 1D-PDPTW instances with 50 and 100 requests by Li and Lim (2001) and found that it reaches nearly the same solution quality of the best PDP(TW) solution methods available. For testing the hybrid 3L-PDP algorithm 54 3L-PDP instances with up to 100 requests and up to 300 boxes were introduced. The results for the three 3L-PDP variants are plausible in that there is a clear tradeoff between travel distance and reloading effort.

In our future research, we will deal with the residual 3L-PDP variants defined here and including the reloading ban, a packing constraint to preclude any reloading effort. In these problem variants, we can expect even better travel distances given the same assumptions in terms of reloading effort.

References

Baldacci, R., Bartolini, E., & Mingozzi, A. (2011). An exact algorithm for the pickup and delivery problem with time windows. *Operations Research, 59*(2), 414–426.

Bartók, T., & Imre, C. (2011). Pickup and delivery vehicle routing with multidimensional loading constraints. *Acta Cybernetica, 20*, 17–33.

Bent, R., & van Hentenryck, P. (2006). A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows. *Computers & Operations Research, 33*, 875–893.

Berbeglia, G., Cordeau, J. F., Gribkovskaia, I., & Laporte, G. (2007). Static pickup and delivery problems: A classification scheme and survey. *Top, 15*, 1–31.

Bortfeldt, A. (2012). A hybrid algorithm for the capacitated vehicle routing problem with three-dimensional loading constraints. *Computers & Operations Research, 39*, 2248–2257.

Bortfeldt, A., & Homberger, J. (2013). Packing first, routing second – A heuristic for the vehicle routing and loading problem. *Computers & Operations Research, 40*, 873–885.

Bortfeldt, A., Hahn, T., Männel, D., & Mönch, L. (2015). Hybrid algorithms for the vehicle routing problem with clustered backhauls and 3D loading constraints. *European Journal of Operational Research, 243*, 82–96.

Brønmo, G., Christiansen, M., Fagerholt, K., & Nygreen, B. (2007). A multi-start local search heuristic for ship scheduling – A computational study. *Computers & Operations Research, 34*, 900–917.

Cordeau, J. F., Iori, M., Laporte, G., & Salazar González, J. J. (2010). Branch-and-cut for the pickup and delivery traveling salesman problem with LIFO loading. *Networks, 55*, 46–59.

Coté, J. F., Gendreau, M., & Potvin, J. Y. (2012). Large neighborhood search for the pickup and delivery traveling salesman problem with multiple stacks. *Networks, 60*, 19–30.

Crainic, T., Perboli, G., & Tadei, R. (2008). Extreme point-based heuristics for three-dimensional bin packing. *INFORMS Journal on Computing, 20*, 368–384.

Derigs, U., & Döhmer, T. (2008). Indirect search for the vehicle routing problem with pickup and delivery and time windows. *OR Spectrum, 30*, 149–165.

Duhamel, C., Lacomme, P., Quilliot, A., & Toussaint, H. (2011). A multi-start evolutionary local search for the two-dimensional loading capacitated vehicle routing problem. *Computers & Operations Research, 38*, 617–640.

Fuellerer, G., Doerner, K. F., Hartl, R., & Iori, M. (2009). Ant colony optimization for the two-dimensional loading vehicle routing problem. *Computers & Operations Research, 36*, 655–673.

Fuellerer, G., Doerner, K. F., Hartl, R., & Iori, M. (2010). Metaheuristics for vehicle routing problems with three-dimensional loading constraints. *European Journal of Operational Research, 201*, 751–759.

Gendreau, M., Iori, M., Laporte, G., & Martello, S. (2006). A tabu search algorithm for a routing and container loading problem. *Transportation Science, 40*, 342–350.

Gendreau, M., Iori, M., Laporte, G., & Martello, S. (2007). A tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. *Networks, 51*, 4–18.

Gendreau, M., & Potvin, J. Y. (2010). *Handbook of metaheuristics* ((2nd ed.)). New York: Springer.

Iori, M., Salazar Gonzalez, J. J., & Vigo, D. (2007). An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transportation Science, 41*, 253–264.

Iori, M., & Martello, S. (2010). Routing problems with loading constraints. *Top, 18*, 4–27.

Iori, M., & Martello, S. (2013). An annotated bibliography of combined routing and loading problems. *Yugoslav Journal of Operations Research, 23*(3), 311–326.

Khebbache-Hadji, S., Prins, C., Yalaoui, A., & Reghioui, M. (2013). Heuristics and memetic algorithm for the two-dimensional loading capacitated vehicle routing problem with time windows. *Central European Journal of Operations Research, 21*, 307–336.

Koning, D. (2011). *Using column generation for the pickup and delivery problem with disturbances. Technical Report*. Department of Computer Science, Utrecht University.

Lacomme, P., Toussaint, H., & Duhamel, C. (2013). A GRASP x ELS for the vehicle routing problem with basic three-dimensional loading constraints. *Engineering Applications of Artificial Intelligence, 26*, 1795–1810.

Leung, S. C. H., Zhang, Z., Zhang, D., Hua, X., & Lim, M. K. (2013). A meta-heuristic algorithm for heterogeneous fleet vehicle routing problems with two-dimensional loading constraints. *European Journal of Operational Research, 225*, 199–210.

Li, H., & Lim, A. (2001). A metaheuristic for the pickup and delivery problem with time windows. In *Proceedings of the 13th IEEE international conference on tools with artificial intelligence (ICTAI'01)* (pp. 333–340). Los Alamitos, CA: IEEE Computer Society.

Lim, H., Lim, A., & Rodrigues, B. (2002). Solving the pickup and delivery problem with time windows using squeaky wheel optimization with local search. In *Proceedings of American conference on information systems, AMICS 2002*. Dallas, USA.

Lu, Q., & Dessouky, M. M. (2006). A new insertion-based construction heuristic for solving the pickup and delivery problem with time windows. *European Journal of Operational Research, 175*, 672–687.

Malapert, A., Guéret, C., Jussien, N., Langevin, A., & Rousseau, L. M. (2008). Two-dimensional pickup and delivery routing problem with loading constraints. In *Proceedings of the first CPAIOR workshop on bin packing and placement constraints (BPPC'08)*. Paris, France.

Moura, A., & Oliveira, J. F. (2009). An integrated approach to vehicle routing and container loading problems. *Operations Research Spectrum, 31*, 775–800.

Nagata, Y., Kobayashi, S., et al. (2010). *A memetic algorithm for the pickup and delivery problem with time windows using selective route exchange crossover*. In R. Schaefer, et al. (Eds.) (pp. 536–545). Berlin: Springer.

Nanry, W. P., & Barnes, W. (2000). Solving the pickup and delivery problem with time windows using reactive tabu search. *Transportation Research Part B, 34*, 107–121.

Pankratz, G. (2005). A grouping algorithm for the pickup and delivery problem with time windows. *OR Spectrum, 27*, 21–41.

Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2008). A survey on pickup and delivery problems. Part II: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft, 58*, 81–117.

Petersen, H. L., & Madsen, O. B. G. (2009). The double travelling salesman problem with multiple stacks – Formulation and heuristic solution approaches. *European Journal of Operational Research, 198*, 139–147.

Pollaris, H., Braekers, K., Caris, A., Janssens, G., & Limbourg, S. (2015). Vehicle routing problems with loading constraints: State-of-the-art and future directions. *OR Spectrum, 37*, 297–330.

Romero, M., Sheremetov, L., & Soriano, A. (2007). A genetic algorithm for the pickup and delivery problem: An application to the helicopter offshore transportation. In O. Castillo (Ed.), *Theoretical advances and applications of fuzzy logic: 42* (pp. 435–444) ASC. Berlin: Springer.

Ropke, S., & Pisinger, D. (2006). An adaptive large neighborhood search for the pickup and delivery problem with time windows. *Transportation Science, 40*, 455–472.

Ropke, S., Cordeau, J. F., & Laporte, G. (2007). Models and a branch-and-cut algorithm for pickup and delivery problems with time windows. *Networks, 49*, 258–272.

Ropke, S., & Cordeau, J. F. (2009). Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science, 43*(3), 267–286.

Ruan, Q., Zhang, Z., Miao, L., & Shen, H. (2013). A hybrid approach for the vehicle routing problem with three-dimensional loading constraints. *Computers & Operations Research, 40*, 1579–1589.

Shaw, P. (1998). Using Constraint programming and local search methods to solve vehicle routing problems. In *Proceedings of the fourth international conference on principles and practice of constraint programming, CP-98*.

Sintef (2015). website of sintef (Norway), Li and Lim Benchmark, www.sintef.no/projectweb/top/pdptw/li-lim-benchmark, last visited 22.04.2016.

Tao, Y., & Wang, F. (2015). An effective tabu search approach with improved loading algorithms for the 3L-CVRP. *Computers & Operations Research, 55*, 127–140.

Tarantilis, C., Zachariadis, E., & Kiranoudis, C. (2009). A hybrid metaheuristic algorithm for the integrated vehicle routing and three-dimensional container-loading problem. *IEEE Transactions on Intelligent Transportation Systems, 10*, 255–271.

Toth, P., & Vigo, D. (2014). *Vehicle routing: problems, methods, and applications* (2nd ed.). Philadelphia: MOS-SIAM series on optimization.

Wang, L., Guo, S., Chen, S., Zhu, W., & Lim, A. (2010). Two natural heuristics for 3D packing with practical loading constraints. *Computer Science, 6230*, 256–267.

Wei, L., Zhang, Z., & Lim, A. (2014). An adaptive variable neighborhood search for a heterogeneous fleet vehicle routing problem with three-dimensional loading constraints. *IEEE Computational Intelligence Magazine, 9*, 18–30.

Wei, L., Zhang, Z., Zhang, D., & Lim, A. (2015). A variable neighborhood search for the capacitated vehicle routing problem with two-dimensional loading constraints. *European Journal of Operational Research, 243*, 798–814.

Wisniewski, M., Ritt, M., & Buriol, L. S. (2011). A tabu algorithm for the capacitated vehicle routing problem with three-dimensional loading constraints. In *Proceedings of anais do XLIII simpósio Brasileiro de pesquisa operacional* (pp. 1502–1511). Ubatuba, Brazil.

Xu, H., Chen, Z. L., Rajagopal, S., & Arunapuram, S. (2003). Solving a practical pickup and delivery problem. *Transportation Science, 37*, 347–364.

Zachariadis, E., Tarantilis, C., & Kiranoudis, C. (2012). The pallet-packing vehicle routing problem. *Transportation Science, 46*, 341–358.

Zachariadis, E., Tarantilis, C., & Kiranoudis, C. (2013). Designing vehicle routes for a mix of different request types, under time windows and loading constraints. *European Journal of Operational Research, 229*, 303–317.

Zhang, Z., Wei, L., & Lim, A. (2015). An evolutionary local search for the capacitated vehicle routing problem minimizing fuel consumption under three-dimensional loading constraints. *Transportation Research Part B, 82*, 20–35.

Zhu, W., Qin, H., Lim, A., & Wang, L. (2012). A two-stage tabu search algorithm with enhanced packing heuristics for the 3L-CVRP and M3L-CVRP. *Computers & Operations Research, 39*, 2178–2195.

Teil III:

Solving the Pickup and Delivery Problem with Three-Dimensional Loading Constraints and Reloading Ban

European Journal of Operational Research 264 (2018) 119–137

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Production, Manufacturing and Logistics

Solving the pickup and delivery problem with three-dimensional loading constraints and reloading ban

Dirk Männel[*], Andreas Bortfeldt

Otto von Guericke University, Universitätsplatz 2, 39106 Magdeburg, Germany

ARTICLE INFO

Article history:
Received 23 August 2016
Accepted 18 May 2017
Available online 25 May 2017

Keywords:
Transportation
Vehicle routing
Packing
Pickup and delivery

ABSTRACT

In this paper, we extend the classical Pickup and Delivery Problem (PDP) to an integrated routing and three-dimensional loading problem, called PDP with three-dimensional loading constraints (3L-PDP). We are given a set of requests and a homogeneous fleet of vehicles. A set of routes of minimum total length has to be determined such that each request is transported from a loading site to the corresponding unloading site. In the 3L-PDP, each request is given as set of rectangular boxes and the vehicle capacity is replaced by a 3D loading space.

This paper is the second one in a series of articles on 3L-PDP. As in the first paper we are dealing with constraints which guarantee that no *re*loading effort will occur. Here the focus is laid on the reloading ban, a packing constraint that ensures identical placements of same boxes in different packing plans. The reloading ban allows for better solutions in terms of travel distance than a routing constraint that was used in the first paper to preclude any reloading effort. To implement this packing constraint a new type of packing procedure is needed that is capable to generate a series of interrelated packing plans per route. This packing procedure, designed as tree search algorithm, and the corresponding concept of packing checks is the main contribution of the paper at hand. The packing procedure and a large neighborhood search procedure for routing form a hybrid algorithm for the 3L-PDP. Computational experiments were performed using 54 3L-PDP benchmark instances.

1. Introduction

This paper is the second one in a series of papers on the pickup and delivery problem with 3D loading constraints (3L-PDP). It continues the paper by Männel and Bortfeldt (2016), henceforth called first paper. We extend the classical PDP to an integrated routing and 3D loading problem. The PDP is defined in Parragh, Doerner, and Hartl (2008); a very recent survey on routing problems with loading constraints can be found in Pollaris et al. (2015). Apart from that we refer the reader to the literature review given in the first paper.

The 3L-PDP can roughly be described as follows. We are given a set of requests and a homogeneous fleet of vehicles. A set of routes, each starting and ending at the single depot, has to be determined such that each request is transported from a loading site to the corresponding unloading site and the total travel distance is minimized. In the 3L-PDP, each request is given as a set of 3D rectangular items (boxes) and the vehicle capacity is replaced by a 3D loading space. Each route has to be completed by a series of packing plans, where a packing plan represents an arrangement of boxes after having visited a pickup or delivery node. Besides basic geometrical constraints (e.g. no overlapping boxes) specific packing constraints as usual in VRP with 3D loading constraints (e.g. support condition) are to be satisfied.

Our main concern in the problem formulation of 3L-PDP is to guarantee that in 3L-PDP solutions any *re*loading effort can be excluded. That is, the boxes should not be moved *after* they were loaded and *before* they are unloaded. In the first paper necessary and sufficient conditions to avoid any reloading effort have been discussed. The results can be summarized as follows:

– It is assumed that boxes are loaded and unloaded at the rear of vehicles. Furthermore, boxes have to be loaded and unloaded by pure movements in length direction as usual in routing problems with 3D loading constraints (see Gendreau et al., 2006).
– First, we must require the request sequence (RS) constraint at delivery and pickup points of a route. At a delivery point, the RS constraint says that between a box A to be unloaded and the rear there is no box B to be unloaded later. Moreover, a box B to be unloaded later must not lie above box A. At a pickup point, the RS constraint requires that between a box A just loaded and

[*] Corresponding author.
E-mail addresses: dirk.maennel@gmx.de (D. Männel), andreas.bortfeldt@ovgu.de (A. Bortfeldt).

http://dx.doi.org/10.1016/j.ejor.2017.05.034
0377-2217/© 2017 Elsevier B.V. All rights reserved.

Table 1
Five 3L-PDP variants (y: yes, n: no, a: automatically).

#	RS loading	RS unloading	Reloading ban	Independent partial routes	Reloading effort	Travel distance
1	y	n	n	n	High	Very low
2	y	y	n	n	Medium	Low
3	y	n	y	n	Medium	Low
4	y	y	y	n	Zero	Medium
5	y	a	a	y	Zero	High

the rear or above box A there is no box B that was loaded at an earlier pickup point. If the RS constraint would not be satisfied at a delivery or pickup point, boxes could not be unloaded or loaded by a pure movement in length direction and without moving other boxes. For a delivery point, placements of other boxes would have to be changed temporarily in order to unload boxes with this destination by pure length shifts. For a pickup point, placements of other boxes must be changed temporarily to reach the final positions for the loaded boxes by pure length movements.

– However, the RS constraint is not sufficient to avoid any reloading effort. In a route for 3L-PDP, generally boxes of a request A are transported for a part of the route together with boxes of a request B and for another part together with boxes of a request C (and no longer with boxes of B) etc. Packing plans have to be provided for all parts of the route in which different sets of boxes are transported. If different packing plans are provided for the boxes of a request A, because the boxes are first to be packed with the boxes of request B and then with the boxes of request C, the placements of the boxes of A may change. This would not necessarily violate required packing constraints. Thus there would exist feasible 3L-PDP solutions including boxes that are to be *re*loaded after loading and before unloading; an elaborated example is shown in the first paper.
– To rule out any reloading effort, we have to specify an extra constraint. There are two options to do so, i.e. we can introduce an additional routing constraint and, alternatively, we can define a packing constraint that rules out any reloading effort.
– The mentioned routing constraint, called independent partial routes (IPR) constraint, rules out any reloading effort by restricting the shape of the routes, i.e. in an implicit fashion. This is done by so-called 3L-PDP routing patterns, which ensure that the boxes of any request are not stored together with boxes of different requests in different parts of a route.
– The additional packing constraint, termed reloading ban, requires that the placement of any box, including the position of a reference corner (or of the geometrical midpoint) and the spatial orientation of the box, must not undergo a (permanent) change after the box has been loaded and before the box is unloaded. The reloading ban is tailored to the general shape of 3L-PDP routes: it forbids explicitly a change of placements of boxes of a request A if they are loaded together with boxes of a request C after they have been loaded together with boxes of a request B.

In the first paper we have introduced a spectrum of five 3L-PDP variants (see Table 1). The RS constraint at pickup points (denoted by (C1)) is always required. The variants are specified by means of the RS constraint for delivery points (C2), the reloading ban (C3) and the IPR constraint (C4). For each variant and constraint the entry is "y" if the constraint is to be met and "n" if not. If the IPR condition and the RS constraint at pickup points are required, RS constraint at delivery points and reloading ban are automatically satisfied; this is marked by entry "a".

We consider variants 4 and 5 as the main 3L-PDP variants. In both variants any reloading effort is excluded by different means, namely by a routing constraint (variant 5) or a packing constraint (variant 4). The main 3L-PDP variants correspond to practical scenarios where a reloading of goods is not a viable option. This can be the case for different reasons, e.g. lack of manpower and equipment or narrow space at customer sites.

However, we also deal with three variants (1–3) where reloading effort is not excluded a priori. In this way we want to study the "costs" of avoidance of reloading effort in terms of travel distance. Generally, we expect a trade-off between travel distance and reloading effort. Thus, in the last two columns of Table 1 the expected reloading effort and expected (total) travel distance are indicated.

Moreover, the problem variants 1–3 might also have some practical relevance. If pickup and delivery transports are to be organized in rural areas with great distances between customers it could be advantageous to accept some reloading effort and to save a large travel distance in return (see Xu, Chen, Rajagopal, & Arunapuram, 2003).

The different 3L-PDP variants are illustrated by some (two-dimensional) single route examples in Fig. 1. The node number 0 denotes the depot while Pi and Di ($i = 1, \ldots , 4$) stand for the pickup and delivery nodes. For all nodes the state of the loading space is shown after the loading/unloading operation at the corresponding node has taken place (view from above). In all examples the driver's cabin is on left and the loading door on the right side of the loading space. In the example for variant 1 for unloading the box I11 at node D1 it is necessary to unload the box I21 first; furthermore the box I21 is reloaded at another position to allow the loading of the boxes I31 and I32 at the following node P3. So neither the RS unloading constraint (C2) nor Reloading ban constraint (C3) is satisfied here. In the example for variant 2 the RS unloading constraint (C2) is satisfied at all delivery nodes but Reloading ban constraint (C3) is not satisfied because the placement of box I12 is changed permanently at P3 to allow the loading of box I31. In the example for variant 3 for unloading the box I11 at node D1 again it is necessary to unload the box I21 first (like in the first example), but now the box I21 is reloaded at the same position, so Reloading ban constraint (C3) is satisfied and only RS unloading constraint (C2) is unsatisfied here. In the example for variant 4 both RS unloading constraint (C2) and Reloading ban constraint (C3) are satisfied, especially the boxes I11 and I12 hold the same placements at nodes P2 and P3 (interrelated packing plans). In the last example for variant 5 the special structure of a route which complies the IPR constraint (C4) is shown. First some pickup points are visited and then all corresponding delivery points follow in inverse order. If one delivery node has been visited a further pickup is only allowed when all boxes are unloaded before. In the example this situation occurs when the vehicle leaves node D3 and goes to pickup node P4.

A precise formulation of the 3L-PDP (including all above variants) with constraints (C1)–(C10) can be found in the first paper. For convenience a short description of constraints (C5)–(C10), not mentioned before, is given in Table 2.

In the first paper the focus was laid on problem variants 1, 2 and 5. *In the second paper we deal with variants 3 and 4.* In variant 5 any reloading effort is excluded by strongly restricting the admitted routes and this will have a negative impact on travel distances. In the 3L-PDP variant 4 there is no restriction of vehicle routes. Instead the reloading ban (C3) is in charge to preclude any reloading effort and better travel distances can be expected. To implement this packing constraint a new type of packing procedure is needed that is capable to generate a series of interrelated packing plans per route (see above example). The design of this packing procedure and the corresponding concept of packing checks is the main contribution of the paper at hand. In the 3L-PDP variant 3 the RS

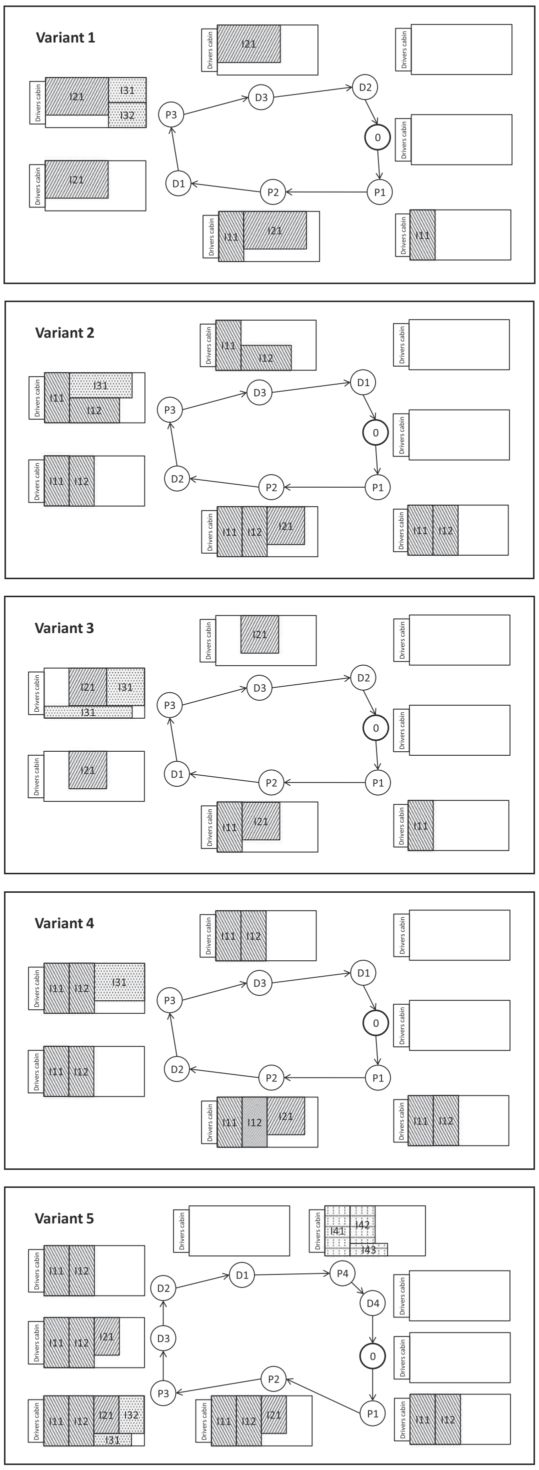


Fig. 1. Examples for 3L-PDP variants.

Table 2
Further constraints of 3L-PDP.

Constraint	Name/short description
(C5)	Weight constraint: stipulates a weight limit of the boxes loaded in a vehicle.
(C6)	Orientation constraint: fixes the height of the boxes; horizontal 90° turns are allowed.
(C7)	Support constraint: stipulates supported percentage of base area of boxes.
(C8)	Stacking constraint: requires that fragile boxes may only bear other fragile boxes.
(C9)	Route length constraint: specifies a maximum allowed route length.
(C10)	Route number constraint: specifies a maximum number of routes.

constraint for delivery points (C2) is not required and this will lead to a certain reloading effort but to even better travel distances than in variant 4.

The rest of the paper is organized as follows: in Section 2, we propose a hybrid algorithm for solving the 3L-PDP with reloading ban. In Section 3 numerical results are presented and in Section 4 the avoidance of reloading effort is considered from a more general perspective. Conclusions are drawn in Section 5.

2. A hybrid algorithm for the 3L-PDP with reloading ban

Our hybrid algorithm for the 3L-PDP with reloading ban is composed of a procedure for routing and one for packing. The routing procedure is derived from the adaptive large neighborhood search (LNS) heuristic for solving the PDP with time windows by Ropke and Pisinger (2006). In fact we use the same routing procedure as in the first paper. The tree search (TRS) algorithm by Bortfeldt (2012) was essentially further developed (for this paper) to specify a packing procedure that is able to observe the reloading ban.

Within each iteration of the LNS procedure for routing, a set of requests is removed from the current solution (s_{curr}) and then reinserted to get the next solution (s_{next}). Afterwards, it is tested whether s_{next} is accepted as new current solution. Otherwise, the previous current solution s_{curr} is kept for the next iteration. Packing checks are primarily carried out in insertion heuristics by which new solutions are generated but are also performed within acceptance checks. In the following sections, the concept of packing checks is established and the packing procedure is described in detail.

2.1. Packing checks

A 3L-PDP solution has to provide feasible packing plans for each route and each visited site per route. The plan for a site must include placements of all boxes already loaded and not yet unloaded after visiting this site. In order to reduce the effort spent for packing checks, we apply a similar methodology as in the first paper to the 3L-PDP variants 3 and 4 with reloading ban. We proceed in four steps:

(1) Additional constraints are formulated that are in general stronger than the RS constraints (C1) and (C2). These are the cumulative request sequence constraints for pickup points (CRS-p) and for delivery points (CRS-dj) ($j = 1, 2$).
(2) It is shown that feasible packing plans for all sites of a route can be derived from feasible packing plans for selected pickup points of this route if the latter plans meet the additional (as well as original) constraints. This is done by means of the Propositions 1 and 2. Thus, the search becomes less costly as independent packing plans are to be provided only for few sites of a route.

Example for problem variant 3 and 4:

Given route

$0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow P3 \rightarrow P4 \rightarrow P5 \rightarrow D5 \rightarrow D4 \rightarrow P6 \rightarrow D3 \rightarrow D6 \rightarrow D1 \rightarrow 0$

Sequence of open pickup points	Packing plan incl. constraints CRS-p CRS-dj (j = 1,2) to be provided for site	Derived packing plans result for sites
1. $P1 \rightarrow P2$	P2	P1, D2
2. $P1 \rightarrow P3 \rightarrow P4 \rightarrow P5$	P5	P3, P4, D5, D4
3. $P1 \rightarrow P3 \rightarrow P6$	P6	D3, D6, D1

- In 3L-PDP variant 3 constraint CRS-d2 has to be met at plans for sites P2, P5 and P6.
- In 3L-PDP variant 4 constraint CRS-d1 has to be met at plans for sites P2, P5 and P6.
- In both variants the packing plans to be provided for P2, P5 and P6
  must include identical placements for boxes of
    - request 1 (occurring in plans for P2, P5 and P6)
    - request 3 (occurring in plans for P5 und P6).

Legend: 0: Depot, Pi / Di: pickup / delivery point of request i, i = 1,...,6.

Fig. 2. Packing checks for 3L-PDP variants 3 and 4.

(3) Then it will be discussed under which circumstances and for which problem variants the search space is reduced by the additional constraints. For this purpose Proposition 3 is proven.

(4) Finally, some differences between 3L-PDP variants 3 and 4 regarding reloading effort are explained.

We define a sequence of open pickup points (SOPP) as a sequence of pickup points within a route of a 3L-PDP solution with following characteristics: (i) the last point of the sequence is followed by a delivery point in the route; (ii) the sequence contains exactly all pickup points of the route whose delivery points lie behind the last sequence point (see Fig. 2 for examples).

Let m_2 ($m_2 \geq 1$) be the number of consecutive pickup points lying at the end of the sequence. Let m_1 ($m_1 \geq 0$) be the number of pickup points that are separated from the last m_2 pickup points by at least one delivery point. Then the SOPP can be denoted as P_i, $i = 1, \ldots, m_1, m_1 + 1, \ldots, m_1 + m_2$ (i.e. P_{m1+m2} is the last point).

We say that a packing plan for pickup point P_{m1+m2} of a SOPP satisfies the cumulative request sequence constraint for pickup points (CRS-p) if the following conditions hold: (i) there are no boxes of a request j (loaded at pickup point P_j) between a box of request i and the rear of the vehicle; (ii) there are no boxes of request j above a box of request i ($i, j = 1, \ldots, m_1 + m_2, j < i$). As shown in the first paper the following proposition holds.

Proposition 1. Let a feasible plan for pickup point P_{m1+m2} of a SOPP exist that meets the constraints (C6)–(C8) and observes the CRS-p constraint. Then feasible packing plans observing constraints (C1) and (C6)–(C8) do also exist for pickup points P_i ($i = m_1 + 1, \ldots, m_1 + m_2 - 1$).

In the first paper, a routing constraint was introduced in order to be able to derive feasible packing plans for the delivery points of a route. In the paper at hand the same purpose is achieved by the following additional packing constraints.

Let D_i be the corresponding delivery points of the pickup points P_i, $i = 1, \ldots, m_1 + m_2$, of a SOPP. We say that a packing plan for pickup point P_{m1+m2} satisfies the cumulative request sequence constraint for delivery points (CRS-d1) if the following conditions hold: (i) if D_i lies before D_j, the boxes of request j must not lie between a box of request i and the rear of the vehicle ($i, j = 1, \ldots, m_1 + m_2$); (ii) under the same assumption, boxes of request j must not lie above a box of request i ($i, j = 1, \ldots, m_1 + m_2$). The constraint (CRS-d2) is defined similarly, but only the second condition (ii) is required.

Proposition 2. Let a SOPP and a packing plan for the last pickup point P_{m1+m2} be given.

(i) If the packing plan is feasible, meets the constraints (C1), (C6)–(C8) and satisfies the CRS-d1 constraint, then feasible packing plans, observing the constraints (C2) and (C6)–(C8), do exist for the consecutive m_3 delivery points behind P_{m1+m2} ($m_3 \geq 1$).

(ii) If constraint CRS-d2 is substituted for CRS-d1, then feasible packing plans, observing constraints (C6)–(C8), do exist for the consecutive m_3 delivery points behind P_{m1+m2} ($m_3 \geq 1$).

Proof. (i) Due to constraint CRS-d1, the boxes for the first delivery point behind P_{m1+m2}, say D_{i1}, are in unloading position in the packing plan for P_{m1+m2}. If the boxes for D_{i1} are removed, a packing plan for D_{i1} results. Since the plan for P_{m1+m2} observes support constraint (C7) and the removed boxes do not support boxes of other requests, the plan for D_{i1} also meets (C7). Constraints (C6) and (C8) as well as feasibility conditions (FP1)–(FP3) hold, as they were met in plan P_{m1+m2}. The boxes for the next delivery point D_{i2} are in unloading position due to constraint CRS-d1, i.e. the plan for D_{i1} also meets constraint (C2). For the following delivery points, packing plans can be derived in a similar manner. (ii) Feasible packing plans, observing constraints (C6)–(C8), for consecutive delivery points behind P_{m1+m2} can be derived as before. In particular, support constraint (C7) holds for these plans due to CRS-d2. □

The constraints CRS-dj ($j = 1, 2$) are formulated for all delivery points that correspond to pickup points P_i, $i = 1, \ldots, m_1 + m_2$, of a given SOPP. However, Proposition 2 only claims that feasible packing plans can be derived for the consecutive delivery points behind P_{m1+m2}. Nevertheless, feasible packing plans can be derived for all delivery points of a route if Proposition 2 is applied to *all* SOPPs of a route. The same holds for the pickup points of a route with regard to Proposition 1.

In 3L-PDP variants 3 and 4, the reloading ban (C3) is required. It forbids that different placements of same boxes occur in packing plans for different sites of a route. We can state that if the reloading ban holds for all packing plans for the last points of SOPPs, then it holds for the derived packing plans for all other pickup and delivery points, too. Therefore, the above results show that for 3L-PDP variants 3 and 4 it is sufficient to construct feasible packing plans for the last pickup points of all sequences of open pickup points in a given route that meet the reloading ban. Feasible packing plans that observe the RS constraints (C1) and (C2) (in case of variant 4) and constraints (C6)–(C8) as well as the reloading ban can then be derived for all other pickup points and all delivery points of this route. Of course, this claim holds only if – for variant 3 – the constraints CRS-p and CRS-d2 are met in the plans for the last pickup points of SOPPs; for variant 4, the constraints CRS-p and CRS-d1 must be observed in these plans.

The procedure of packing checks for a route in 3L-PDP variants 3 and 4 is illustrated by an example in Fig. 2.

As said above we want to discuss the question whether the additional constraints CRS-*p* and CRS-d*j* ($j = 1, 2$) will reduce the search space, i.e. the set of all feasible solutions. For this purpose the following proposition is shown.

Proposition 3. Let a SOPP with $m_1 + m_2$ points be given and let the reloading ban (C3) be required.

(i) Let feasible packing plans exist for all pickup points P_i of the SOPP ($i = 1, \dots, m_1 + m_2$). The plans should meet constraint (C1). Then the packing plan for point P_{m1+m2} observes constraint CRS-*p*.

(ii) Let feasible packing plans exist for all delivery points D_i that correspond to pickup points P_i of the SOPP ($i = 1, \dots, m_1 + m_2$). The plans should meet constraint (C2). Then the packing plan for point P_{m1+m2} observes constraint CRS-d1. If the plans for the delivery points obey the condition that any box to be unloaded does not lie under a box to be unloaded later then the packing plan for P_{m1+m2} observes at least constraint CRS-d2.

Proof. (i) Suppose constraint CRS-p1 is violated in the packing plan of last pickup point P_{m1+m2}. Then two requests and corresponding pickup points *i* and *j* ($1 \le j < i \le m_1 + m_2$) must exist so that there is a box of request *j* that lies between a box of request *i* and the rear or above that box. Due to the reloading ban the box of request *i* would not be in unloading position within packing plan for pickup point *i*, contradictory to constraint (C1).

(ii) For all delivery points D_i ($i = 1, \dots, m_1 + m_2$) the boxes to be unloaded are in unloading position due to (C2). The reloading ban guarantees that the placements of these boxes are the same in the packing plan for pickup point P_{m1+m2}. Thus, constraint CRS-d1 – being just the sum of the (C2) requirements for multiple delivery points – is observed in the plan for P_{m1+m2}. The last assertion results in an analog manner. □

For problem variants 3 and 4 the reloading ban (C3) is required. Hence, it can be concluded that the additional constraints in fact do not reduce the search space. The same applies for problem variant 5 (dealt with in the first paper as well as problem variants 1 and 2) since in this variant the IPR constraint (C4) is required that implies the reloading ban. It can be easily seen that the reloading ban (C3) is an indispensable assumption in Proposition 3. Hence, for problem variants 1 and 2 the additional constraints cause in general a reduction of the search space since constraint (C3) is not assumed.

In 3L-PDP variant 4, there is no reloading effort at all, while in variant 3 some reloading effort can occur at delivery sites. If a vehicle arrives at a delivery site, all boxes of the corresponding request, say *A*, are to be unloaded. Since RS constraint (C2) is not required, some boxes of requests *B*, *C*, etc. may stand in the way of the *A*-boxes. These are called blocking boxes. We assume that blocking boxes have to be temporarily unloaded (and blocking boxes of blocking boxes, etc.). Because of constraint CRS-d2, blocking boxes cannot occur above boxes to be unloaded. For this reason, temporarily unloaded boxes can afterwards be loaded again so that they take their original placements.

There is no significant difference between packing checks performed in insertion heuristics and those performed in acceptance checks, i.e. in any case for a given route the necessary feasible packing plans for last pickup points of SOPPs are to be provided.

2.2. Packing procedure

The packing procedure should be able to implement the reloading ban (C3). Packing plans that are generated for the last pickup points of SOPPs in a route need to be interrelated if the SOPPs have a common pickup point. That is, if the same boxes are stowed in more than one of these packing plans, their placements must coincide. To ensure this, the packing plans for a route with multiple SOPPs are generated at once, i.e. by means of one and the same depth first search.

For the depth first search, a route is organized in multiple pickup and delivery sequences (PDS). A PDS contains the last m_2 ($m_2 \ge 1$) consecutive pickup points of a SOPP and the following m_3 ($m_3 \ge 1$) consecutive delivery points. A route consists of several PDSs and a packing plan is needed for each of these PDS, i.e. for its last pickup point. Fig. 3 shows the PDS related to the SOPPs in Fig. 2.

The depth first search is carried out by means of the recursive procedure extend_packing_plan (see Algorithm 1) and the subordinated procedure initialize_packing_state (see Algorithm 2).

The PDSs are indexed by *ipds* and the set *freeBoxes* includes the boxes of a PDS that are still available; *ipds* is set to zero and *freeBoxes* is set empty before the first call of the recursive procedure. The set *potentialPlacements* comprises potential placements of boxes in *freeBoxes*. Implemented placements for the current PDS are collected in the set *PDSPlan*, while the complete solution with the placements of all PDSs is held in the set *totalPlan*.

In procedure extend_packing_plan it is checked first whether the set *freeBoxes* is empty, i.e. whether the packing plan for the current PDS is complete. In this case (and if *ipds* > 0) this plan is incorporated in the complete solution *totalPlan*. The placements of boxes to be unloaded at delivery sites of the current PDS are marked in *totalPlan*.

Afterwards index *ipds* is incremented and procedure initialize_packing_state is called for the new PDS. The complete solution *totalPlan* is only initialized empty for *ipds* $= 1$. The set *freeBoxes* is reinitialized and then includes the boxes that belong to the PDS. Potential placements for the whole set of boxes of current PDS in the lower left front corner of the loading space $L \times W \times H$ (at the driver's cabin) are generated.

Then all placements, already put in *totalPlan* and *not* marked as unloaded, are reinserted in the new PDS solution *PDSPlan*. Each time another "old" placement is reinserted, the set *potentialPlacements* is updated taking into account all already inserted placements. After the *for*-loop is executed the current solution *PDSPlan* is filled with all placements of former PDSs that remain placements of the present PDS. As these placements are copied it is ensured that placements of same boxes in different PDS coincide. At the same time the set *potentialPlacements* at the end comprises only such placements which are compatible with all these "old" placements.

The current instance of procedure extend_packing_plan is aborted if there is at least one free box without a potential placement, i.e. if a complete solution can no longer be achieved on this search path. Candidates for the next placement for PDS *ipds* are selected from list *potentialPlacements* and are provided in the list *currentPlacements*. All these placements are then tried alternatively. For each placement, the current PDS solution, the set of free boxes

Examples for pickup and delivery sequences (PDS):

Given route

$0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow P3 \rightarrow P4 \rightarrow P5 \rightarrow D5 \rightarrow D4 \rightarrow P6 \rightarrow D3 \rightarrow D6 \rightarrow D1 \rightarrow 0$

Sequence of open pickup points	Related PDS
1. $P1 \rightarrow P2$	$P1 \rightarrow P2 \rightarrow D2$
2. $P1 \rightarrow P3 \rightarrow P4 \rightarrow P5$	$P3 \rightarrow P4 \rightarrow P5 \rightarrow D5 \rightarrow D4$
3. $P1 \rightarrow P3 \rightarrow P6$	$P6 \rightarrow D3 \rightarrow D6 \rightarrow D1$

Legend: 0: Depot, Pi / Di: pickup / delivery point of request i, i = 1,...,6.

Fig. 3. Examples of PDS.

Algorithm 1 Packing procedure 1, extend_packing_plan.

```
extend_packing_plan (inout: ipds, freeBoxes, potentialPlacements, PDSPlan, totalPlan)
if number of procedure calls > maxApCalls then abort packing check endif
if freeBoxes=∅ then
      if ipds > 0 then
            totalPlan:= totalPlan ∪ PDSPlan
            mark all placements in totalPlan as unloaded whose boxes belong
              to requests with delivery sites in PDS(ipds)
      endif
      ipds:= ipds + 1                                          // next PDS
      if totalPlan complete then abort packing check endif
      initialize_packing_state(ipds, freeBoxes, potentialPlacements, PDSPlan, totalPlan)
Endif
if there is at least one box in freeBoxes(ipds) without placement in potentialPlacements then return endif
provide list currentPlacements with potential placements that are currently to be tried
for i:= 1 to |currentPlacements| do
      PDSPlan':= PDSPlan ∪ {currentPlacements(i)}              // add placement to PDS-plan
      freeBoxes':= freeBoxes \ {currentPlacements(i).box}      // update free boxes
      potentialPlacements':= update(potentialPlacements)       // update potential placements
      extend_packing_plan (ipds, PDSPlan', freeBoxes', potentialPlacements', totalPlan)    // recursive call
endfor
end.
```

Algorithm 2 Packing procedure 2, initialize_packing_state.

```
initialize_packing_state (in: ipds, out: freeBoxes, potentialPlacements, PDSPlan, inout: totalPlan)
if ipds = 1 then totalPlan=∅ endif
freeBoxes:= {boxes to be loaded in PDS ipds}
initialize set potentialPlacements for box set freeBoxes and empty loading space
PDSPlan:= ∅
for all placements Pl in totalPlan in given loading order not marked as unloaded do
      PDSPlan:= PDSPlan ∪ {Pl}
      potentialPlacements:= update(potentialPlacements)
endfor
end.
```

and the set of potential placements are updated accordingly, before procedure extend_packing_plan is called again. To update the list *potentialPlacements,* all potential placements are removed that can no longer be implemented. Additional potential reference points for new placements are determined as extreme points (see Crainic, Perboli, & Tadei, 2008).

The selection of placements currently to be tried among all potential placements is governed by two rules. On the one hand, it is ensured that a vehicle is loaded from the front to the back, from bottom to top with lower priority, and from left to right with lowest priority. Hence, placements with smaller *x*-coordinates of the reference corner are preferred, etc. On the other hand, placements of boxes are preferred that belong to earlier loaded requests and, therefore, have to be stowed nearer to the cabin. The placement selection is controlled by the integer parameters *maxBoxRankDiff* and *maxRefPoints* where higher parameter values lead to a larger set of currently tried placements (see Bortfeldt, 2012).

Potential placements are generated and updated in such a way that the packing plan for a PDS (i.e. for the last pickup point of the corresponding SOPP) is feasible and observes constraints (C6)–(C8), CRS-p and CRS-d1 (variant 4) or CRS-d2 (variant 3).

The packing check for a route is terminated when the number of recursive calls of procedure extend_packing_plan exceeds the specified limit *maxApCalls* or *totalPlan* is complete, i.e. a solution containing placements for all boxes to be loaded and unloaded in a given route is reached.

A cache of tested request sequences (routes) is used to accelerate the search as described in the first paper.

In the following example, given for the 2L-PDP, we illustrate how the packing procedure is checking a route and generating interrelated packing plans. In Fig. 4 the route and the items to be loaded and unloaded, the SOPPs and related PDSs are shown. Two interrelated packing plans are needed for the last pickup points of the SOPPs, namely P2 and P3. The first packing plan must include the items of requests 1 and 2, the second plan has to contain the

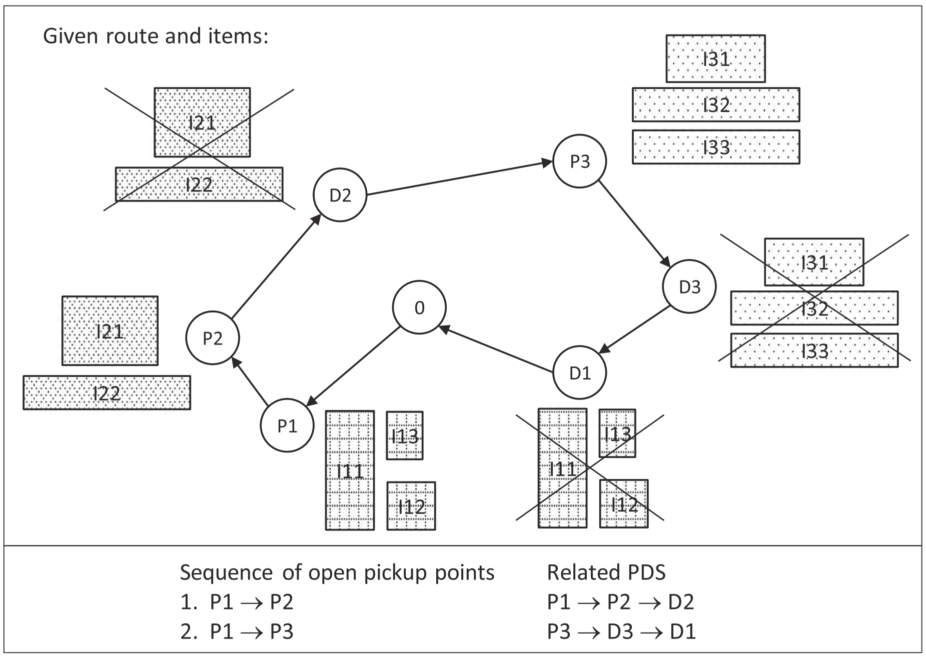


Fig. 4. Illustration of packing procedure (input).

items of requests 1 and 3. Moreover, the placements of items of request 1 must coincide in both plans to observe the reloading ban (C3).

In Fig. 5 it is shown how the search runs until feasible packing plans for P2 and P3 are reached by means of eight consecutive search states. The states are commented as follows:

State 1 (P2):

- after *ipds* is set to 1 the set *freeBoxes* is initialized as {I11, I12, I13, I21, I22} and potential placements for the empty loading space are generated;
- after placements were selected for the 3 boxes of P1 and for box I21 it turns out in the 5th recursive call of extend_packing_plan that there is no (feasible) placement for box I22;
- this negative test result is illustrated (here as in following states) by an *in*feasible placement for I22 and the phrase “not ok”.

State 2 (P2):

- after selecting a next placement for box I21 there is again no feasible placement for box I22.

State 3 (P2, P3):

- after selecting further placements for boxes I13 and I21 now a feasible placement for box I22 is found; thus, a feasible packing plan for P2 is reached (indicated by phrase “ok”) and set freeBoxes becomes empty;
- the five placements in *PDSplan* are added to *totalPlan* and the boxes I21 and I22 are labelled as unloaded;
- now *ipds* is set to 2 and *freeBoxes* is reinitialized as {I31, I32, I33}; *PDSplan* is filled by the placements of boxes I11, I12 and I13 that were calculated earlier and *potentialPlacements* is initialized accordingly; in Fig. 5 the initial state of *PDSplan* is shown for *ipds* = 2 (P3).

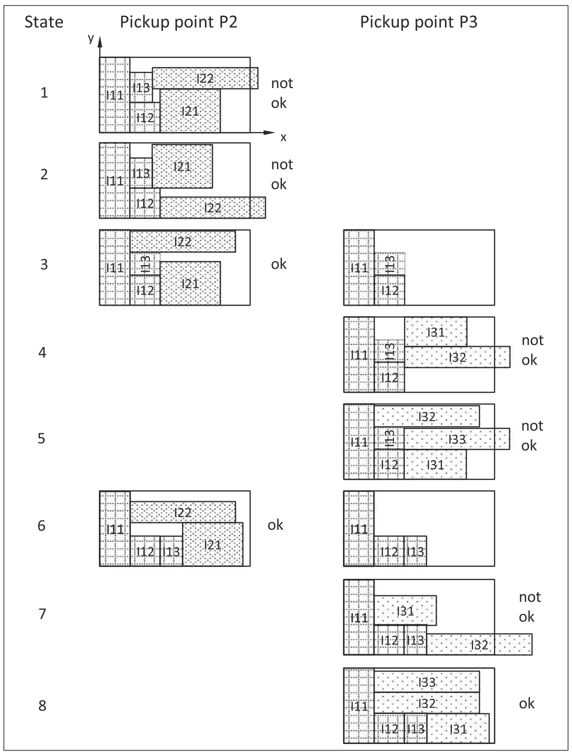


Fig. 5. Illustration of packing procedure (search states, top view).

State 4 (P3):

– after selecting a placement for box I31 there is no feasible placement for box I32 in the following recursive call.

State 5 (P3):

– after selecting an alternative placement for box I31 now a feasible placement for box I32 is found; however, there is no feasible placement for box I33;
– we suppose that further placements for boxes I31 and I32 do not exist; hence the search jumps back to the state where alternatives regarding the chosen placement of box I13 are tried; at the same time we have again *ipds* $=1$, *freeBoxes*$=\{$I13, I21, I22$\}$ etc.

State 6 (P2, P3):

– after an alternative placement for box I13 has been chosen the boxes I21 and I22 can also be placed; thus another feasible packing plan for pickup point P2 is reached;
– the further proceeding is as in state 3.

State 7 (P3):

– after selecting a placement for box I31 there is no feasible placement for box I32.

State 8 (P3):

– after selecting an alternative placement for box I31 the boxes I32 and I33 can be feasibly placed, too;
– hence a feasible packing plan is reached for pickup point P3; the packing solution *totalPlan* for the given route is complete and the search is aborted;
– the placements of the boxes I11, I12 and I13 coincide in both packing plans for P2 and P3, i.e. the reloading ban (C3) is observed.

Fig. 5 also illustrates how (potential) placements are generated. Free boxes are to be placed at such locations where they cannot be moved in negative x-, y- and z-direction. Note that in a given search state potential placements for more than one box and/or location might be provided (not shown in Fig. 5 for the sake of convenience).

3. Computational experiments

In the computational experiments we test the hybrid algorithm for both "new" 3L-PDP variants 3 and 4 and the "old" variants 1A, 1B, 2 and 5. The subvariants 1A and 1B differ from each other regarding the reloading policy for temporarily unloaded boxes. In the first policy (subvariant 1A), the original loading order of all requests is maintained. In the second policy (subvariant 1B), the original loading order is modified as the order of requests for which temporarily unloaded boxes do exist is now chosen inverse to the order of corresponding delivery points. Moreover, we will hybridize the algorithm variants 3 and 5 as well as 4 and 5 in order to reduce the necessary packing effort and to generate high quality solutions quicker (see below).

The experiments are carried out using the 54 3L-PDP instances with up to 100 requests and up to 300 boxes introduced in the first paper. In total 30 instances with 50 requests, 18 instances with 75 and 6 instances with 100 requests are provided. The average number of boxes per request is two for half and three for the other half of the instances. Regarding the distribution of pickup and delivery points of the requests three variants are distinguished:

- RAND ("Random"): the sites are uniformly distributed in a rectangular section of the plane,
- CLUS ("Mixed Cluster"): the sites are clustered, each cluster may contain pickup as well as delivery points,
- CPCD ("Pure Cluster"): the sites are clustered, only sites of one sort can occur in each cluster.

Table 3
Computing time limits in minutes for experiments.

Number of requests	2 boxes per request on avg.	3 boxes per request on avg.
50	5	10
75	10	20
100	20	40

The dimensions of the uniform loading spaces of the vehicles are chosen as $L=60$, $W=25$ and $H=30$ length units. The lengths, widths and heights of the boxes are chosen randomly from intervals that range from 20% to 60% percent of the length, width and height, respectively, of the loading spaces. A box is characterized as fragile with the probability 0.25. The percentage a for the minimal supporting area was specified as 0.75.

The packing procedure is coded in the C++ programming language using Visual Studio 2012 Express, while the LNS scheme is implemented using the Java programming language (version 6u35, 32 bit) under Eclipse 3.5.2. Preliminary experiments (in which total run times were varied) demonstrated that the impact of the different developing environments is negligible. All the experiments have been conducted on a PC with Intel Core i5-2500 K (4.0 gigahertz, 16 gigabytes) running Windows 10 Professional. The java virtual machine uses a maximum heap size of 2 gigabytes memory.

The parameter setting is taken over from the first paper, however, the parameter *maxApCalls* (see Section 2.2) is set to 3000 if the checked route matches the IPR constraint (C4); otherwise the parameter is set to 200. If (C4) is met by a route, the effort for packing checks is much lower than in the opposite case and the checks can be carried out more intensively, i.e. with a higher value of *maxApCalls*. If (C4) is not met, this high value would require too much packing effort and so the value *maxApCalls* $=200$ is taken. In Table 3 the maximum run time per instance and single run is shown. The computing time depends on the number of requests and the average box number per request.

3.1. Results for 3L-PDP variants 4 and 4*

The results for the 3L-PDP instances regarding total travel distance (*ttd*) with no reloading effort are presented in Table 4. In the first two columns the instance type and the number of instances are listed. The next column shows the total travel distances for 3L-PDP (or algorithm) variant 5 where all constraints including the *request sequence constraint* for both pickup and delivery points (C1) and (C2) and the *independent partial routes constraint* (C4) are considered. In the following four columns total travel distances and gaps are indicated for the 3L-PDP variants 4 and 4* (see Table 1). In variant 4, the IPR constraint (C4) is not considered and the reloading effort is ruled out by the weaker the *reloading ban constraint* (C3). In the additional algorithm variant 4*, the variant 4 is hybridized with variant 5: in the first 40% of the computing time the algorithm has to construct routes which respect the IPR constraint (C4), i.e. it behaves as variant 5. Because the effort for packing checks strongly depends on the form of the routes, the algorithm can make much more iterations in the same time if it is restricted to IPR-routes because they are much easier to check than Non-IPR-routes.

All presented total travel distances are mean values over five runs. To keep the tables compact the results are averaged furthermore over all instances of the same type, e.g. "75_RAND_3" stands

Table 4
Results (travel distances) for 3L-PDP variants without reloading.

Instance type	Number of	Variant 5	Variant 4		Variant 4*	
	instances	ttd	ttd	Gap (%)	ttd	Gap (%)
50_RAND_2	5	1630.19	1539.35	−5.54	1530.48	−6.10
50_CLUS_2	5	1205.73	1150.11	−4.68	1147.24	−4.91
50_CPCD_2	5	1328.63	1317.81	−0.83	1304.78	−1.84
50_RAND_3	5	1617.09	1525.99	−5.60	1523.82	−5.73
50_CLUS_3	5	1169.13	1125.32	−3.84	1122.74	−4.06
50_CPCD_3	5	1312.73	1319.99	0.56	1313.82	0.05
75_RAND_2	3	2139.53	2083.22	−2.63	2052.05	−4.09
75_CLUS_2	3	1461.22	1423.36	−2.59	1405.99	−3.77
75_CPCD_2	3	2237.07	2211.52	−1.15	2195.18	−1.87
75_RAND_3	3	2106.97	2073.10	−1.61	2036.24	−3.37
75_CLUS_3	3	1450.20	1443.27	−0.49	1423.34	−1.86
75_CPCD_3	3	2209.82	2249.65	1.82	2220.03	0.47
100_RAND_2	1	4054.89	3991.39	−1.57	3947.88	−2.64
100_CLUS_2	1	4178.58	4130.58	−1.15	4036.36	−3.40
100_CPCD_2	1	4259.89	4272.27	0.29	4278.91	0.45
100_RAND_3	1	3995.01	4044.72	1.24	4004.29	0.23
100_CLUS_3	1	4100.42	4149.49	1.20	4102.87	0.06
100_CPCD_3	1	4201.28	4320.01	2.83	4203.64	0.06
Sum	**54**		**Average gap**	**−2.16**		**−2.99**

Table 5
Average gap for small, midsize and large instances.

Number of requests	Variant 4 average gap in %	Variant 4* average gap in %
50	−3.32	−3.77
75	−1.11	−2.41
100	0.47	−0.87

Table 6
Average gap for "random", "mixed cluster" and "pure cluster" instances.

Type	Variant 4 average gap in %	Variant 4* average gap in %
Random	−3.82	−4.66
Mixed cluster	−2.88	−3.62
Pure cluster	0.21	−0.70

for instances of distribution type "Random" with 75 requests and 3 boxes per request on average. The corresponding gaps are calculated as $(ttd - ttd_{V5})$ / ttd_{V5} * 100 (%) (ttd_{V5} stands for ttd in variant 5). In the last line of Table 4 the gap values of the 3L-PDP variants are averaged over the 54 instances.

Detailed results for each single instance with regard to travel distances and other aspects are presented in Tables 15–20 of Appendix A. Algorithm variant 4 achieves a mean reduction of total travel distance by 2.16% compared to variant 5, while 4* reaches a reduction of 2.99%. The hybridized algorithm variants 4*, in which the search is temporarily restricted to IPR-routes, turns out to be rather successful and performs 0.83%-points better than variant 4.

Tables 5 and 6 indicate the influence of instance size and type (regarding distribution of sites) on the solution quality for the algorithm variants 4 and 4*. For small instances with up to 50 requests variant 4 achieves significant better results than variant 5, while for large instances with 100 requests variant 5 performs slightly better than variant 4. However, variant 4*, which temporarily restricts the search space, shows its strength just for large and difficult instances and performs better than variant 5. On the other hand, the difference between variant 4* and 4 is almost negligible for small instances.

With regard to the instance types "Random", "Mixed cluster" and "Pure cluster" it can be observed that variant 4 yields largest improvements compared with variant 5 for instance type "Random" and provides smallest improvements for type "Pure cluster".

Table 7
Results of the test of different IPR-usage values in variant 4*.

IPR-usage (in %)	Average gap to variant 4	Average share of IPR-routes in the best solution (in %)
0	0.000	3.26
30	−0.585	4.85
40	−0.716	5.56
50	−0.693	8.44
60	−0.689	9.65

Again, a significant improvement of results is achieved by the hybridized variant 4* and it is reached especially for the "problematic" instance type "Pure cluster".

The value of 40% IPR-usage in variant 4* was determined in advance when different values of this parameter where tested against each other over three runs per instance. The results of this preliminary test are shown in Table 7. The value of 40% IPR-usage performs better the values of 30%, 50% and 60%, nevertheless the differences are small, especially between the three values of 40%, 50% and 60%. In the last column the average percentage of IPR-routes in the best solutions are shown. As expected the plain variant 4 (IPR-usage $= 0$) realizes the lowest share of IPR-routes (3.26%), while the highest share of 9.65% is realized with IPR-usage= 60%.

3.2. Analysis of trade-off between travel distance and reloading effort

In the following we deal with the tradeoff between travel distance and reloading quantity. In variants 4, 4* and 5 there is no reloading effort as the *Reloading ban* (C3) is in force (see Table 1). Among the variants with Reloading ban variant 4* provides the best results in terms of total travel distance. Thus, variant 4* will be compared now with 3L-PDP algorithm variants 1A, 1B, 2, 3 and 3* regarding total travel distance and reloading effort. Again the additional algorithm variant 3* is the result of a hybridization of variant 3 and 5: in the first 40% of the computing time the algorithm has to construct routes which respect the IPR constraint (C4), i.e. it behaves as variant 5. Table 8 is organized as Table 4 and shows the total travel distances and gaps (as percentages) based on variant 4*.

The reloading effort needed for a 3L-PDP instance is primarily given as reloading quantity, i.e. as the weight of all boxes that are reloaded. If a box is reloaded, say, two times, the weight of the box is counted two times. Thus it may occur that the reloading quantity exceeds the total weight of the boxes. Table 9 is organized as follows. The first two columns include the instance type and the number of instances while the third column shows the total weight of all requests per instance (cargo weight). In the following ten columns the reloading quantities for the relevant 3L-PDP variants are given as absolute values (in weight units) and as percentages of the cargo weight. The results are again averaged over five runs per instance and over all instances of same type. In the last line the percentaged reloading quantities are averaged over the 54 instances. Since the reloading effort is zero for problem variant 4*, this variant does not occur in Table 9.

The reloading quantities of variant 2 (missing Reloading ban) and 3/3* (missing RS constraint for delivery points) are moderate and amount to 13.13% and 25.97%/24.13% of the cargo weight on average. For problem variants 1A and 1B, where both constraints are missing, the mean reloading quantity is much higher (97.27% and 84.24%, respectively). However, the variants 2 and 3/3* bring only a small decrease of the total travel distance (0.82% and 0.65%/1.36%) while the variants 1A and 1B reduce the total travel distance much stronger (9.29% and 9.06%). Again the hybridized variant 3* beats the original variant 3 (0.71 %-point less ttd and 1.84 %-point less reloading quantity). Table 10 summarizes the

Table 8
Results (travel distances) for 3L-PDP variants with reloading.

Instance	Number of	Variant 4*	Variant 3		Variant 3*		Variant 2		Variant 1A		Variant 1B	
type	instances	ttd	ttd	Gap (%)	ttd	Gap (%)	ttd	Gap (%)	ttd	Gap (%)	ttd	Gap (%)
50_RAND_2	5	1530.48	1498.33	−2.08	1507.08	−1.52	1556.05	1.70	1349.69	−11.82	1354.05	−11.52
50_CLUS_2	5	1147.24	1131.22	−1.37	1125.91	−1.84	1133.72	−1.16	1041.13	−9.33	1043.36	−9.13
50_CPCD_2	5	1304.78	1311.57	0.53	1287.68	−1.30	1285.73	−1.44	1222.76	−6.32	1227.09	−5.98
50_RAND_3	5	1523.82	1492.86	−2.03	1494.67	−1.93	1542.00	1.19	1343.85	−11.82	1352.04	−11.30
50_CLUS_3	5	1122.74	1114.46	−0.75	1108.66	−1.27	1113.96	−0.70	1032.61	−8.11	1035.71	−7.80
50_CPCD_3	5	1313.82	1302.55	−0.85	1297.66	−1.22	1287.91	−1.95	1236.04	−5.98	1234.50	−6.07
75_RAND_2	3	2052.05	2030.96	−1.03	2009.15	−2.08	2057.41	0.27	1781.93	−13.17	1789.72	−12.79
75_CLUS_2	3	1405.99	1406.01	−0.01	1384.46	−1.54	1396.88	−0.65	1281.86	−8.83	1290.89	−8.19
75_CPCD_2	3	2195.18	2180.95	−0.65	2160.47	−1.58	2169.99	−1.15	2012.82	−8.31	2027.72	−7.64
75_RAND_3	3	2036.24	2018.90	−0.86	2011.06	−1.24	2001.99	−1.68	1777.74	−12.72	1776.47	−12.77
75_CLUS_3	3	1423.34	1447.78	1.70	1425.42	0.14	1393.35	−2.09	1280.43	−10.05	1280.27	−10.05
75_CPCD_3	3	2220.03	2195.87	−1.08	2190.69	−1.31	2159.45	−2.72	2056.00	−7.37	2056.40	−7.35
100_RAND_2	1	3947.88	3907.97	−1.01	3869.08	−2.00	3988.25	1.02	3476.54	−11.94	3470.88	−12.08
100_CLUS_2	1	4036.36	4168.36	3.27	4015.00	−0.53	3994.20	−1.04	3614.50	−10.45	3672.62	−9.01
100_CPCD_2	1	4278.91	4276.60	−0.05	4316.77	0.88	4190.00	−2.08	4092.40	−4.36	4114.89	−3.83
100_RAND_3	1	4004.29	3899.90	−2.61	3861.05	−3.58	3938.97	−1.63	3455.99	−13.69	3428.00	−14.39
100_CLUS_3	1	4102.87	4170.34	1.64	4081.65	−0.52	3951.76	−3.68	3659.60	−10.80	3632.85	−11.46
100_CPCD_3	1	4203.64	4302.57	2.35	4215.58	0.28	4163.11	−0.96	4109.32	−2.24	4067.89	−3.23
Sum	**54**	**Average gap**		**−0.65**		**−1.36**		**−0.82**		**−9.29**		**−9.06**

Table 9
Reloading quantities for 3L-PDP variants with reloading.

Instance type	Number of instances	Cargo weight	Variant 3 Reloading quantity		Variant 3* Reloading quantity		Variant 2 Reloading quantity		Variant 1A Reloading quantity		Variant 1B Reloading quantity	
			Absolute	In %	Absolute	In %	Absolute	In %	Absolute	In %	Absolute	In %
50_RAND_2	5	614051	139349	23.20	118517	19.55	82736	13.67	567811	94.05	523557	86.58
50_CLUS_2	5	614051	136304	22.69	130123	21.62	124195	21.06	542193	89.88	463839	76.63
50_CPCD_2	5	614051	157793	25.92	138684	22.88	42247	7.14	463367	76.54	393270	64.85
50_RAND_3	5	614939	167622	27.61	153770	25.51	85148	14.39	555439	91.95	512386	84.65
50_CLUS_3	5	614939	142066	23.81	141701	23.89	76647	12.56	538931	88.52	486807	80.02
50_CPCD_3	5	614939	156576	25.76	154133	25.52	43735	7.33	493698	81.61	401100	65.77
75_RAND_2	3	787000	212497	26.99	192199	24.36	118068	15.04	849065	107.89	765390	97.24
75_CLUS_2	3	787000	194717	24.71	174115	22.11	131709	16.76	919464	116.79	729781	92.67
75_CPCD_2	3	787000	176909	22.50	183829	23.38	68569	8.73	750783	95.39	660077	83.83
75_RAND_3	3	788876	230248	29.14	211637	26.78	113555	14.42	898871	113.95	795780	100.87
75_CLUS_3	3	788876	194528	24.64	205283	25.97	109747	13.99	918785	116.38	801924	101.58
75_CPCD_3	3	788876	257841	32.69	233459	29.64	77676	9.91	803098	101.89	662256	84.03
100_RAND_2	1	1072407	332020	30.96	277317	25.86	122109	11.39	1275821	118.97	1169467	109.05
100_CLUS_2	1	1072407	266633	24.86	277927	25.92	216976	20.23	1285608	119.88	1099816	102.56
100_CPCD_2	1	1072407	285790	26.65	187515	17.49	199586	18.61	1199016	111.81	899469	83.87
100_RAND_3	1	1074809	364044	33.87	405311	37.71	102135	9.50	1313254	122.18	1074617	99.98
100_CLUS_3	1	1074809	334929	31.16	290559	27.03	189203	17.60	1301508	121.09	1073135	99.84
100_CPCD_3	1	1074809	298355	27.76	190291	17.70	156586	14.57	955683	88.92	866282	80.60
Sum	**54**		**Average**	**25.97**		**24.13**		**13.13**		**97.27**		**84.24**

Table 10
Tradeoff between total travel distance and reloading quantity.

3L-PDP variant	Total travel distance in %	Reloading quantity average in %
5	100.00	0.00
4	97.84	0.00
4*	97.01	0.00
3	96.38	25.97
3*	95.68	24.13
2	96.21	13.13
1A	87.99	97.27
1B	88.21	84.24

results regarding total travel distance and reloading effort. For each 3L-PDP variant the total travel distance is now given as percentage of the travel distance of variant 5 while the reloading quantities are again indicated as percentages of the cargo weight. All presented values are averaged over the five runs per instance and over the 54 3L-PDP instances. The indicated figures for the 3L-PDP variants correspond very well with the expected differences between those variants regarding travel distances and reloading effort as shown in Table 1.

3.3. Analysis of categories of reloading effort

In the following we analyze the reloading effort in greater detail. In generally there exist three reasons to reload a certain box at a certain point:

- a box must be temporarily removed from the vehicle because it's blocking an unloading operation at a delivery point of box(es) belonging to the request of this delivery point ("blocking box"),
- a box occupies in the packing plan of this point another position in loading space than in the last packing plan before ("repositioned box"),
- a box occupies in the packing plan of this point the same position in loading space with changed orientation compared to the last packing plan before ("rotated box").

In Table 11 it is shown how the total reloading effort splits up in three categories for the variants 1A, 1B and 2. For each instance

Table 11
Categories of reloading effort for 3L-PDP variants 1A, 1B and 2.

Instance type	Number of instances	Variant 2 Reloading quantity in %			Variant 1A Reloading quantity in %				Variant 1B Reloading quantity in %			
		Total	Repos	Rotate	Total	Block	Repos	Rotate	Total	Block	Repos	Rotate
50_RAND_2	5	13.67	11.99	1.68	94.05	61.81	30.75	1.49	86.58	54.07	30.73	1.78
50_CLUS_2	5	21.06	18.98	2.08	89.88	61.20	27.70	0.98	76.63	46.18	28.51	1.94
50_CPCD_2	5	7.14	5.86	1.27	76.54	50.66	24.93	0.95	64.85	39.79	23.88	1.18
50_RAND_3	5	14.39	13.23	1.17	91.95	67.11	23.98	0.86	84.65	59.08	24.56	1.01
50_CLUS_3	5	12.56	11.92	0.65	88.52	62.68	24.35	1.48	80.02	53.32	24.68	2.02
50_CPCD_3	5	7.33	6.51	0.82	81.61	58.92	21.68	1.01	65.77	44.15	20.36	1.25
75_RAND_2	3	15.04	14.29	0.75	107.89	75.29	30.64	1.96	97.24	61.92	33.33	2.00
75_CLUS_2	3	16.76	14.48	2.28	116.79	77.79	36.81	2.20	92.67	59.88	30.64	2.15
75_CPCD_2	3	8.73	7.33	1.40	95.39	63.63	28.68	3.08	83.83	47.21	33.04	3.58
75_RAND_3	3	14.42	12.64	1.78	113.95	83.60	29.47	0.88	100.87	68.97	30.31	1.59
75_CLUS_3	3	13.99	13.29	0.70	116.38	84.81	30.09	1.47	101.58	68.46	31.61	1.50
75_CPCD_3	3	9.91	9.16	0.75	101.89	72.63	27.76	1.49	84.03	55.08	27.76	1.19
100_RAND_2	1	11.39	10.20	1.18	118.97	79.52	37.93	1.52	109.05	65.93	40.03	3.09
100_CLUS_2	1	20.23	18.22	2.01	119.88	83.64	33.86	2.38	102.56	63.87	36.21	2.48
100_CPCD_2	1	18.61	16.29	2.32	111.81	68.25	41.20	2.36	83.87	46.67	35.19	2.02
100_RAND_3	1	9.50	7.88	1.63	122.18	95.48	24.99	1.72	99.98	69.74	27.91	2.34
100_CLUS_3	1	17.60	16.02	1.58	121.09	89.37	30.30	1.42	99.84	65.94	32.25	1.66
100_CPCD_3	1	14.57	12.75	1.82	88.92	63.05	24.51	1.35	80.60	51.65	26.18	2.77
Average		**13.13**	**11.80**	**1.33**	**97.27**	**67.86**	**27.97**	**1.44**	**84.24**	**54.28**	**28.18**	**1.78**

Table 12
Maximal loaded volumes for different 3L-PDP variants.

Instance	Number of	Variant 5		Variant 4*		Variant 3*		Variant 2		Variant 1A		Variant 1B	
type	instances	Volume	%	Volume	%	Volume	%	Volume	%	Volume	%	Volume	%
50_RAND_2	5	25808	57.35	24396	54.21	25563	56.81	26469	58.82	28739	63.87	28257	62.79
50_CLUS_2	5	27458	61.02	27380	60.84	27328	60.73	27170	60.38	28080	62.40	29110	64.69
50_CPCD_2	5	28848	64.11	28992	64.43	28474	63.28	29175	64.83	30184	67.08	30081	66.85
50_RAND_3	5	26677	59.28	25922	57.60	26030	57.85	27092	60.20	27894	61.99	27958	62.13
50_CLUS_3	5	26211	58.25	26200	58.22	26590	59.09	27362	60.80	28396	63.10	28086	62.41
50_CPCD_3	5	28899	64.22	29204	64.90	28587	63.53	29132	64.74	29128	64.73	29311	65.14
75_RAND_2	3	27760	61.69	27170	60.38	27307	60.68	28410	63.13	29005	64.46	29011	64.47
75_CLUS_2	3	28916	64.26	28216	62.70	28242	62.76	28714	63.81	30290	67.31	29621	65.83
75_CPCD_2	3	29261	65.02	28534	63.41	28803	64.01	29333	65.18	29630	65.84	29851	66.34
75_RAND_3	3	27337	60.75	26368	58.59	25068	55.71	27264	60.59	28534	63.41	27846	61.88
75_CLUS_3	3	26555	59.01	26755	59.46	26287	58.42	27505	61.12	28016	62.26	28123	62.49
75_CPCD_3	3	28196	62.66	27706	61.57	28041	62.31	28393	63.10	29229	64.95	29835	66.30
100_RAND_2	1	27759	61.69	27996	62.21	27160	60.36	28997	64.44	29828	66.28	28472	63.27
100_CLUS_2	1	31157	69.24	27745	61.66	27712	61.58	29865	66.37	29864	66.36	31206	69.35
100_CPCD_2	1	31101	69.11	30356	67.46	29602	65.78	30307	67.35	29741	66.09	30309	67.35
100_RAND_3	1	29233	64.96	26428	58.73	25816	57.37	27798	61.77	29635	65.86	28920	64.27
100_CLUS_3	1	28683	63.74	27729	61.62	27704	61.56	29229	64.95	28589	63.53	28722	63.83
100_CPCD_3	1	29084	64.63	27713	61.58	28987	64.42	28581	63.51	29187	64.86	28554	63.45
Average			**61.75**		**60.60**		**60.54**		**62.37**		**64.32**		**64.32**

type and each variant the total reloading quantity (as in Table 9 before) and the reloading efforts in the categories "blocking box", "repositioned box" and "rotated box" are shown (all values are percentages of the cargo weight). If a box gets unloaded at a delivery point because it is blocking an unloading operation and afterwards it is reloaded at another position or in another direction this box is only counted in category "blocking". Generally the reloading effort in category "rotated box" is small and amounts to approximately only 5–10% of the reloading effort in category "repositioned box". For variants 1A and 1B the reloading effort in category "blocking box" is nearly twice as much as the reloading effort in the other categories, while in variant 2 no reloading effort of type "blocking box" exists because constraints (C1) and (C2) are in force. Furthermore in variants 3 and 3* with active reloading ban constraint (C3) all reloading effort belongs to category "blocking box". That is why these variants are not included in Table 11. In this paper we refrain from presenting reloading costs, but with the listed reloading effort values it would be easily possible to weight the three categories with different coefficients and calculate a total reloading cost value.

3.4. Analysis of capacity utilization

In Table 12 the maximal volume of contained boxes is shown which the vehicles are reaching within their tours. The maximal loaded volumes are averaged over the routes and over five runs per instance and are given as absolute values and as percentages of the loading space volume (45,000) for each instance type. For all problem variants (variant 3 and 4 are omitted here for sake of simplicity) an average maximum space utilization greater 60% is reached. Very similar utilizations are reported for advanced 3L-CVRP methods, e.g. Bortfeldt (2012), for the benchmark instances proposed by Gendreau et al. (2006). Since the boxes and loading spaces in our 3L-PDP instances are constructed exactly in the same fashion our utilization rates seem to be quite good, although the results of 3L-CVRP and 3L-PDP are not completely comparable because of the different problem structure. Generally we expect that the relaxation of constraints (C2) (RS constraint at delivery points) and (C3) (Reloading ban) leads to better space utilizations. The results of variants 1A, 1B and 2 confirm this assumption. The maximum loaded volume for variant 2 (62.37%) and for variants 1A

Table 13
Total iteration numbers and computing times to find the best solution for different 3L-PDP variants.

Instance	CPU	Variant 5		Variant 4^*		Variant 3^*		Variant 2		Variant 1A		Variant 1B	
type	seconds	Total iterations	Runtime to best in %	Total iterations	Runtime to best in %	Total iterations	Runtime to best in %	Total iterations	Runtime to best in %	Total iterations	Runtime to best in %	Total iterations	Runtime to best in %
50_RAND_2	300	179863	49.04	49378	73.77	47696	69.19	99106	39.05	14762	69.82	11374	61.27
50_CLUS_2	300	215667	33.98	59301	69.99	57354	82.83	100539	49.41	13638	69.14	9997	76.19
50_CPCD_2	300	90092	59.72	8463	76.44	8714	74.92	31266	69.44	3223	70.56	2478	76.68
50_RAND_3	600	129033	51.02	12035	76.83	10425	79.86	90373	50.97	6627	75.86	4499	80.53
50_CLUS_3	600	214049	49.94	9924	83.28	10387	80.22	120305	50.68	3947	84.77	2424	80.29
50_CPCD_3	600	12607	85.85	1610	74.56	1465	82.81	4518	80.56	793	80.66	690	79.64
75_RAND_2	600	132369	47.22	21826	72.52	24684	83.10	59991	50.99	4728	79.46	3252	81.52
75_CLUS_2	600	138430	54.84	19993	85.35	19425	85.88	44476	65.61	3696	81.85	2726	80.03
75_CPCD_2	600	63513	68.07	8702	81.91	8738	84.08	24366	83.26	3012	82.67	2553	76.67
75_RAND_3	1200	44377	73.91	4071	78.25	4080	82.07	32464	68.85	1652	84.80	1707	79.08
75_CLUS_3	1200	69441	73.63	3479	76.22	3016	79.25	13009	77.86	1387	89.15	1178	88.74
75_CPCD_3	1200	14643	86.42	2630	68.43	2498	82.31	6833	84.36	1034	82.53	1014	79.01
100_RAND_2	1200	81845	60.78	9222	86.11	9639	85.23	37656	74.47	3271	77.07	3018	86.51
100_CLUS_2	1200	94320	71.72	14982	83.06	14242	78.51	49142	62.38	2804	79.36	2112	73.07
100_CPCD_2	1200	6599	84.08	2205	81.17	2154	68.84	4138	80.63	1369	77.42	1133	69.73
100_RAND_3	2400	28326	89.91	3382	83.47	3262	76.17	11727	89.93	1459	85.00	1392	88.66
100_CLUS_3	2400	32224	86.26	3247	77.53	3136	89.88	10669	86.85	1234	70.12	1012	76.92
100_CPCD_3	2400	3430	89.43	1450	72.91	1379	79.19	2636	82.81	753	84.58	719	88.66
Average		**108,178**	**61.89**	**17,040**	**76.79**	**16,691**	**79.95**	**53,517**	**64.27**	**5044**	**78.31**	**3777**	**77.99**

and 1B (64.32%) is larger than for the other variants. For the variants 3^* and 4^* (which use the new packing algorithm to cope with the reloading ban) the maximum loaded volume is slightly smaller than for variant 5, so there is probably still potential for improvement of the new packing algorithm.

3.5. Analysis of computational effort

In Table 13 the average total numbers of iterations executed and the average computing times to find the best solution are shown. The times are given as percentages of the allowed computing time per instance type shown in the second column. All values are averaged over five runs and over all instances of the same type. The results show that the "old" variant 5 only needs 61.89% of the computing time on average to find the best solutions while the "new" variants 4^* and 3^* need 76.79% and 79.95%, respectively, to find the best solutions. Furthermore in variant 5 around six times more iterations can be executed than in variant 4^* and 3^* in the same available computing time (108,178 vs. 17,040 and 16,691). This shows that the "new" variants are more expensive in terms of CPU usage because of the more complex packing algorithm. On the other hand, there may be still potential for further improvements with the "new" variants if more CPU time would be allowed. The total number of iterations executed for variants 1A and 1B is quite low compared to variants 2 and 5. In variants 1A and 1B there are much more possibilities to construct routes because of the missing constraints (C2)–(C4). Thus the algorithm makes more packing checks of *different* routes in these variants while in variants 2 and 5 the algorithm often repeats packing checks of the *same* routes. Since the packing cache speeds up the repetitions of packing checks significantly, in variants 2 and 5 much more iterations can be executed in the same time.

4. On the avoidance of reloading effort for different vehicle routing problems

In the following, we want to deal with the question of avoiding reloading effort from a more general perspective. We take different basic vehicle routing problems (VRP), as considered in Toth and Vigo (2014), combined with 3D loading constraints and ask for packing constraints which prevent any reloading effort. Moreover, we want to determine specific demands on packing algorithms, related to the avoidance of reloading effort, for different types of VRP. We consider VRP with a single depot. As before we assume that vehicles are rear-loaded and only horizontal shifts of boxes in length direction of a vehicle are allowed for loading and unloading operations.

In Table 14 the results for different basic VRP types with 3D loading constraints are summarized. We differentiate the capacitated VRP (3L-CVRP), the distance constraint VRP (3L-DCVRP) and the VRP with time windows (3L-VRPTW). Furthermore, the VRP with clustered backhauls (3L-VRPCB), as most simple VRP with backhauls, is considered. In the 3L-VRPCB each customer is either a linehaul or a backhaul customer and in each route all linehaul customers must be delivered with goods from the depot before goods can be picked up at backhaul customers. For the 3L-PDP two variants are taken into account. In the first variant the independent partial routes (IPR) constraint is required and cares for the

Table 14
Avoidance of reloading effort for different VRPs.

3L-VRP	*Packing constraints for avoidance of reloading effort*			*Demands on packing algorithm*	
	Request sequence at delivery points	Request sequence at pickup points	Reloading ban	No. of packing plans per route	Interrelated packing plans
3L-CVRP	✓			1	
3L-DCVRP	✓			1	
3L-VRPTW	✓			1	
3L-VRPCB	✓	✓		2	
3L-PDP					
– with IPR constraint		✓		>1	
– without IPR constraint	✓	✓	✓	>1	✓

avoidance of reloading effort (see first paper). In the second variant the IPR constraint is not demanded.

For the 3L-CVRP a request sequence constraint at delivery points, alias LIFO (last-in first-out) policy, is sufficient to rule out any reloading effort (see, e.g., Gendreau et al., 2006 and constraint (C2) in the first paper). Obviously, the same applies to the 3L-DCVRP and 3L-VRPTW. For all three problems only one packing plan per route has to be provided. The plan must include a feasible arrangement of all boxes that belong to the customers visited on this route. It shows the state of the vehicle loading space when the vehicle departures from the depot. The LIFO constraint ensures that feasible packing arrangements result for all visited sites on that route by successive unloading of boxes.

In the 3L-VRPCB a request sequence constraint at delivery points (LIFO) and at pickup points (FILO, first-in last-out policy) is needed (see Bortfeldt et al., 2015, and constraint (C1) in the first paper). Two packing plans have to be provided per route in general. The first plan shows the loading state of the vehicle when departing from depot and must include all boxes of the visited linehaul customers. The second plan shows the loading state of the vehicle when arriving again at the depot and must contain all boxes picked up at the backhaul customers. Again, from these two plans feasible box arrangements can be derived for all visited sites in the route.

For the 3L-PDP with additional IPR constraint (C4), dealt with in the first paper, only the request sequence constraint at pickup points is necessary to prevent any reloading effort. The request sequence constraint at delivery points is then satisfied automatically. By the way, the reverse statement is also true: if IPR constraint is observed then the request sequence constraint for pickup points is satisfied automatically if the request sequence constraint for delivery points is respected. Generally, multiple packing plans are to be provided with a route. Each of these packing plans must contain the boxes of some consecutive pickup points that are included in a sub-pattern of a route and are followed by the related delivery points in reverse order. Feasible packing arrangements can then be derived for all pickup and delivery points of this sub-pattern.

In the 3L-PDP without IPR constraint we must demand the request sequence constraint in both variants and the reloading ban (C3) to prevent any reloading effort. A packing plan is needed for each sequence open pickup points (SOPP, see definition in Section 2.1) of a route. The new phenomenon is here that a pickup point can occur in multiple SOPPs (see Fig. 2). In order to observe the reloading ban, the placements of the boxes of a pickup point that belongs to two SOPPs must coincide in related packing plans. Hence, the packing plans for a route can in general no longer be generated independently of each other. This is a new requirement in 3L-VRP that has not been encountered yet. Up to now it was sufficient to employ a packing algorithm for the 3L-CVRP (able to observe the LIFO constraint) and to apply this algorithm several times per route if necessary. For example, in the 3L-VRPCB such an algorithm has to be used two times per route for a packing check. Thus the 3L-PDP without IPR constraint stands for a new level of difficulty of packing checks.

All in all, more intricate 3L-VRPs require more complex restrictions to avoid reloading effort and more difficult packing algorithms to perform the needed packing checks.

In practical scenarios often further packing constraints have to be satisfied besides those considered here. Weight distribution and horizontal stability constraints can be mentioned as prominent examples (see Bortfeldt & Wäscher, 2013). Weight distribution constraints assure that, regarding the 3L-PDP, the weight of the cargo is spread evenly over the floor of the loading space. Horizontal stability constraints require that the boxes cannot shift significantly in the loading space when the vehicle is traveling. Horizontal stability is often measured by the percentage of boxes that are sufficiently laterally supported by other boxes or the side walls of the loading space (Bischoff & Ratcliff, 1995).

With regard to the 3L-PDP it is important to note that weight distribution and horizontal stability measures should be applied at all customer sites of a route, whenever the loading space is loaded or unloaded. If necessary the packing plan has to be adapted, i.e. a reloading of goods should take place. For example, to maintain a sufficiently even weight distribution it might be necessary to shift the load in length direction. Obviously, this type of reloading would have a positive effect as it helps to observe given constraints. And this makes the difference to the reloading of goods considered in the paper at hand and excluded by the reloading ban constraint. The reloading considered here simply occurs when the connection between packing plans for different sites of a route remains disregarded.

It remains a topic of future research to take into account further constraints (as the above mentioned) and to avoid unnecessary reloading as well as to allow for necessary reloading of goods.

5. Conclusions

In the paper at hand and in the previous paper by Männel and Bortfeldt (2016), the vehicle routing problem with pickup and delivery (PDP) has been extended to an integrated vehicle routing and loading problem (3L-PDP). In the problem formulation we concentrated on the question under which conditions any reloading effort, i.e. any movement of boxes *after* loading and *before* unloading, can be avoided. It turned out that the request sequence constraints (C1) and (C2) for pickup and delivery points are not sufficient. Instead, we must require either a new routing constraint, called independent partial routes condition (C4), or a new packing constraint, termed reloading ban (C3), to exclude any reloading effort. Eventually, a spectrum of five 3L-PDP variants was introduced that allow for different portions of reloading effort and reciprocal savings of travel distance.

In this paper, we focused on the reloading ban, a packing constraint that ensures identical placements of same boxes in different packing plans. A hybrid algorithm for solving the 3L-PDP with reloading ban consisting of a routing and a packing procedure has been proposed. As in the first paper the routing procedure performs a large neighborhood search. A tree search heuristic is responsible for packing boxes that stems from the packing procedure published by Bortfeldt (2012). However, to cope with the reloading ban, i.e. to generate interrelated packing plans for a given route, the packing procedure was substantially enhanced.

In detail, the hybrid algorithm was developed here for problem variants 4 and 3 (see Table 1). Moreover, the corresponding algorithm variants were hybridized with algorithm variant 5 (from the first paper) for the 3L-PDP with independent partial routes condition resulting in algorithm variants 4* and 3*. All new and old variants of the hybrid algorithm were tested by means of 54 3L-PDP instances with up to 100 requests and up to 300 boxes. The new variants 4 and 4* for the 3L-PDP without any permitted reloading effort reached noticeable smaller travel distances compared with the rival variant 5. The improvement in terms of travel distance amounts to 2.2% for algorithm variant 4 and to 3.0% for variant 4*. The comparison of the variants of the hybrid algorithm shows a clear tradeoff between travel distance and reloading effort and confirms the theoretical expectations. All algorithm variants reached maximal volume utilizations per tour above 60% on average and this can be evaluated as a rather good result. Finally, the reloading effort has been differentiated where necessary in the categories “blocking”, “repositioning” and “rotating”. While blocking of boxes is the major source of reloading a considerable share of reloading might also be caused by violations of the reloading ban.

Future research on 3L-PDP should consider, as discussed in Section 4, further constraints which are indispensable requirements in practice.

Appendix A

Tables 15–20.

Table 15
Travel distances for 3L-PDP variants without reloading (complete results).

Instance	Variant 5	Variant 4		Variant 4*	
	ttd	ttd	Gap (%)	ttd	Gap (%)
50_RAND_2_1	1736.90	1635.72	−5.83	1632.39	−6.02
50_RAND_2_2	1585.41	1506.50	−4.98	1498.12	−5.51
50_RAND_2_3	1654.94	1526.49	−7.76	1522.29	−8.02
50_RAND_2_4	1579.04	1540.53	−2.44	1507.85	−4.51
50_RAND_2_5	1594.68	1487.54	−6.72	1491.74	−6.46
50_CLUS_2_1	1121.67	1071.01	−4.52	1066.95	−4.88
50_CLUS_2_2	1108.00	1036.82	−6.42	1042.52	−5.91
50_CLUS_2_3	1150.45	1092.75	−5.02	1081.49	−5.99
50_CLUS_2_4	1276.50	1229.84	−3.66	1231.86	−3.50
50_CLUS_2_5	1372.05	1320.13	−3.78	1313.36	−4.28
50_CPCD_2_1	1366.02	1347.40	−1.36	1348.35	−1.29
50_CPCD_2_2	1257.05	1270.08	1.04	1238.81	−1.45
50_CPCD_2_3	1233.61	1201.01	−2.64	1190.61	−3.49
50_CPCD_2_4	1330.88	1318.11	−0.96	1302.37	−2.14
50_CPCD_2_5	1455.59	1452.44	−0.22	1443.77	−0.81
50_RAND_3_1	1719.96	1591.50	−7.47	1595.12	−7.26
50_RAND_3_2	1562.69	1463.68	−6.34	1477.23	−5.47
50_RAND_3_3	1649.81	1553.59	−5.83	1542.63	−6.50
50_RAND_3_4	1564.77	1507.59	−3.65	1493.06	−4.58
50_RAND_3_5	1588.24	1513.57	−4.70	1511.07	−4.86
50_CLUS_3_1	1049.84	1026.93	−2.18	1025.09	−2.36
50_CLUS_3_2	1097.03	1009.62	−7.97	1009.61	−7.97
50_CLUS_3_3	1124.03	1075.38	−4.33	1067.27	−5.05
50_CLUS_3_4	1250.32	1213.93	−2.91	1213.40	−2.95
50_CLUS_3_5	1324.44	1300.73	−1.79	1298.32	−1.97
50_CPCD_3_1	1336.16	1353.83	1.32	1344.86	0.65
50_CPCD_3_2	1243.28	1273.55	2.43	1244.26	0.08
50_CPCD_3_3	1239.31	1220.64	−1.51	1221.48	−1.44
50_CPCD_3_4	1310.50	1317.87	0.56	1310.83	0.03
50_CPCD_3_5	1434.41	1434.08	−0.02	1447.64	0.92
75_RAND_2_1	2127.32	2097.80	−1.39	2064.44	−2.96
75_RAND_2_2	2118.42	2052.71	−3.10	2014.83	−4.89
75_RAND_2_3	2172.86	2099.16	−3.39	2076.89	−4.42
75_CLUS_2_1	1465.26	1429.87	−2.42	1402.13	−4.31
75_CLUS_2_2	1423.63	1385.28	−2.69	1384.34	−2.76
75_CLUS_2_3	1494.76	1454.94	−2.66	1431.49	−4.23
75_CPCD_2_1	2220.77	2185.09	−1.61	2184.97	−1.61
75_CPCD_2_2	2215.28	2184.20	−1.40	2181.01	−1.55
75_CPCD_2_3	2275.16	2265.27	−0.43	2219.57	−2.44
75_RAND_3_1	2137.18	2135.84	−0.06	2076.73	−2.83
75_RAND_3_2	2066.38	2031.04	−1.71	1978.53	−4.25
75_RAND_3_3	2117.35	2052.43	−3.07	2053.47	−3.02
75_CLUS_3_1	1452.80	1448.22	−0.32	1427.99	−1.71
75_CLUS_3_2	1426.90	1400.21	−1.87	1391.35	−2.49
75_CLUS_3_3	1470.90	1481.37	0.71	1450.67	−1.38
75_CPCD_3_1	2217.96	2243.73	1.16	2242.87	1.12
75_CPCD_3_2	2181.64	2265.49	3.84	2224.54	1.97
75_CPCD_3_3	2229.88	2239.72	0.44	2192.67	−1.67
100_RAND_2_1	4054.89	3991.39	−1.57	3947.88	−2.64
100_CLUS_2_1	4178.58	4130.58	−1.15	4036.36	−3.40
100_CPCD_2_1	4259.89	4272.27	0.29	4278.91	0.45
100_RAND_3_1	3995.01	4044.72	1.24	4004.29	0.23
100_CLUS_3_1	4100.42	4149.49	1.20	4102.87	0.06
100_CPCD_3_1	4201.28	4320.01	2.83	4203.64	0.06
Average gap			**−2.16**		**−2.99**

Table 16
Travel distances for 3L-PDP variants with reloading (complete results).

Instance	Variant 4*	Variant 3		Variant 3*		Variant 3*		Variant 1A		Variant 1B	
	ttd	ttd	Gap (%)	ttd	Gap (%)	ttd	Gap (%)	ttd	Gap (%)	ttd	Gap (%)
50_RAND_2_1	1632.39	1580.50	−3.18	1593.55	−2.38	1628.98	−0.21	1451.72	−11.07	1443.78	−11.55
50_RAND_2_2	1498.12	1473.70	−1.63	1478.31	−1.32	1509.92	0.79	1296.90	−13.43	1311.25	−12.47
50_RAND_2_3	1522.29	1494.39	−1.83	1513.11	−0.60	1565.60	2.84	1327.22	−12.81	1327.24	−12.81
50_RAND_2_4	1507.85	1482.04	−1.71	1488.77	−1.27	1536.88	1.92	1338.72	−11.22	1351.54	−10.37
50_RAND_2_5	1491.74	1461.04	−2.06	1461.63	−2.02	1538.87	3.16	1333.91	−10.58	1336.47	−10.41
50_CLUS_2_1	1066.95	1058.89	−0.76	1056.09	−1.02	1033.56	−3.13	973.56	−8.75	970.99	−8.99
50_CLUS_2_2	1042.52	1021.88	−1.98	1017.66	−2.38	1031.54	−1.05	922.95	−11.47	924.61	−11.31
50_CLUS_2_3	1081.49	1076.99	−0.42	1068.08	−1.24	1097.76	1.50	982.26	−9.18	993.78	−8.11
50_CLUS_2_4	1231.86	1203.49	−2.30	1196.04	−2.91	1207.84	−1.95	1102.98	−10.46	1105.22	−10.28
50_CLUS_2_5	1313.36	1294.85	−1.41	1291.68	−1.65	1297.92	−1.18	1223.90	−6.81	1222.21	−6.94
50_CPCD_2_1	1348.35	1354.34	0.44	1332.60	−1.17	1299.20	−3.64	1245.40	−7.64	1234.03	−8.48
50_CPCD_2_2	1238.81	1258.25	1.57	1223.39	−1.24	1223.62	−1.23	1150.80	−7.10	1150.01	−7.17
50_CPCD_2_3	1190.61	1192.47	0.16	1175.53	−1.27	1181.09	−0.80	1108.01	−6.94	1121.75	−5.78
50_CPCD_2_4	1302.37	1302.11	−0.02	1287.82	−1.12	1298.78	−0.28	1242.12	−4.63	1246.26	−4.31
50_CPCD_2_5	1443.77	1450.71	0.48	1419.08	−1.71	1425.95	−1.23	1367.47	−5.29	1383.38	−4.18
50_RAND_3_1	1595.12	1564.84	−1.90	1586.98	−0.51	1609.61	0.91	1437.77	−9.86	1457.05	−8.66
50_RAND_3_2	1477.23	1447.36	−2.02	1436.80	−2.74	1465.56	−0.79	1286.62	−12.90	1293.19	−12.46
50_RAND_3_3	1542.63	1523.62	−1.23	1507.37	−2.29	1559.11	1.07	1317.93	−14.57	1338.00	−13.26
50_RAND_3_4	1493.06	1474.05	−1.27	1489.26	−0.25	1539.78	3.13	1336.28	−10.50	1330.80	−10.87
50_RAND_3_5	1511.07	1454.43	−3.75	1452.97	−3.85	1535.92	1.64	1340.68	−11.28	1341.14	−11.25
50_CLUS_3_1	1025.09	1022.42	−0.26	1014.84	−1.00	1006.31	−1.83	960.04	−6.35	957.98	−6.55
50_CLUS_3_2	1009.61	996.39	−1.31	995.10	−1.44	1024.23	1.45	901.92	−10.67	914.94	−9.38
50_CLUS_3_3	1067.27	1058.69	−0.80	1048.66	−1.74	1073.48	0.58	975.84	−8.57	982.76	−7.92
50_CLUS_3_4	1213.40	1197.67	−1.30	1195.03	−1.51	1185.00	−2.34	1102.77	−9.12	1095.18	−9.74
50_CLUS_3_5	1298.32	1297.11	−0.09	1289.67	−0.67	1280.80	−1.35	1222.49	−5.84	1227.70	−5.44
50_CPCD_3_1	1344.86	1327.71	−1.28	1331.64	−0.98	1317.75	−2.02	1269.26	−5.62	1258.34	−6.43
50_CPCD_3_2	1244.26	1239.97	−0.35	1238.47	−0.47	1226.44	−1.43	1160.73	−6.71	1175.74	−5.51
50_CPCD_3_3	1221.48	1206.45	−1.23	1198.29	−1.90	1199.48	−1.80	1127.45	−7.70	1114.17	−8.79
50_CPCD_3_4	1310.83	1310.19	−0.05	1296.66	−1.08	1288.48	−1.70	1242.54	−5.21	1258.95	−3.96
50_CPCD_3_5	1447.64	1428.40	−1.33	1423.24	−1.69	1407.42	−2.78	1380.22	−4.66	1365.29	−5.69
75_RAND_2_1	2064.44	2073.08	0.42	2024.95	−1.91	2046.09	−0.89	1820.99	−11.79	1827.68	−11.47
75_RAND_2_2	2014.83	1992.09	−1.13	1984.15	−1.52	2034.26	0.96	1733.08	−13.98	1745.30	−13.38
75_RAND_2_3	2076.89	2027.72	−2.37	2018.35	−2.82	2091.87	0.72	1791.71	−13.73	1796.17	−13.52
75_CLUS_2_1	1402.13	1411.63	0.68	1386.25	−1.13	1392.63	−0.68	1301.54	−7.17	1294.09	−7.71
75_CLUS_2_2	1384.34	1359.54	−1.79	1353.69	−2.21	1369.47	−1.07	1242.68	−10.23	1266.05	−8.54
75_CLUS_2_3	1431.49	1446.86	1.07	1413.43	−1.26	1428.53	−0.21	1301.37	−9.09	1312.54	−8.31
75_CPCD_2_1	2184.97	2162.85	−1.01	2140.11	−2.05	2137.11	−2.19	1980.47	−9.36	1979.54	−9.40
75_CPCD_2_2	2181.01	2177.48	−0.16	2150.61	−1.39	2163.29	−0.81	1996.02	−8.48	2012.59	−7.72
75_CPCD_2_3	2219.57	2202.53	−0.77	2190.70	−1.30	2209.57	−0.45	2061.97	−7.10	2091.03	−5.79
75_RAND_3_1	2076.73	2093.40	0.80	2053.37	−1.12	2029.52	−2.27	1870.63	−9.92	1859.06	−10.48
75_RAND_3_2	1978.53	1961.37	−0.87	1954.68	−1.21	1949.88	−1.45	1699.55	−14.10	1712.45	−13.45
75_RAND_3_3	2053.47	2001.95	−2.51	2025.13	−1.38	2026.56	−1.31	1763.04	−14.14	1757.92	−14.39
75_CLUS_3_1	1427.99	1468.09	2.81	1431.95	0.28	1390.36	−2.64	1315.63	−7.87	1302.01	−8.82
75_CLUS_3_2	1391.35	1397.03	0.41	1385.54	−0.42	1379.31	−0.87	1226.99	−11.81	1251.55	−10.05
75_CLUS_3_3	1450.67	1478.21	1.90	1458.77	0.56	1410.38	−2.78	1298.66	−10.48	1287.25	−11.27
75_CPCD_3_1	2242.87	2179.47	−2.83	2190.66	−2.33	2152.01	−4.05	2032.12	−9.40	2037.96	−9.14
75_CPCD_3_2	2224.54	2209.73	−0.67	2171.75	−2.37	2152.32	−3.25	2040.13	−8.29	2029.52	−8.77
75_CPCD_3_3	2192.67	2198.40	0.26	2209.67	0.78	2174.01	−0.85	2095.74	−4.42	2101.73	−4.15
100_RAND_2_1	3947.88	3907.97	−1.01	3869.08	−2.00	3988.25	1.02	3476.54	−11.94	3470.88	−12.08
100_CLUS_2_1	4036.36	4168.36	3.27	4015.00	−0.53	3994.20	−1.04	3614.50	−10.45	3672.62	−9.01
100_CPCD_2_1	4278.91	4276.60	−0.05	4316.77	0.88	4190.00	−2.08	4092.40	−4.36	4114.89	−3.83
100_RAND_3_1	4004.29	3899.90	−2.61	3861.05	−3.58	3938.97	−1.63	3455.99	−13.69	3428.00	−14.39
100_CLUS_3_1	4102.87	4170.34	1.64	4081.65	−0.52	3951.76	−3.68	3659.60	−10.80	3632.85	−11.46
100_CPCD_3_1	4203.64	4302.57	2.35	4215.58	0.28	4163.11	−0.96	4109.32	−2.24	4067.89	−3.23
Average gap			**−0.65**		**−1.36**		**−0.82**		**−9.29**		**−9.06**

Table 17
Reloading quantities for 3L-PDP variants with reloading (complete results).

Instance	Cargo weight	Variant 3 Reloading quantity		Variant 3* Reloading quantity		Variant 2 Reloading quantity		Variant 1A Reloading quantity		Variant 1B Reloading quantity	
		Absolute	In %	Absolute	In %	Absolute	In %	Absolute	In %	Absolute	In %
50_RAND_2_1	610544	133685	21.90	122379	20.04	85412	13.99	500179	81.92	470990	77.14
50_RAND_2_2	578322	123924	21.43	95667	16.54	97065	16.78	622609	107.66	576804	99.74
50_RAND_2_3	530415	180784	34.08	133350	25.14	81630	15.39	646899	121.96	574922	108.39
50_RAND_2_4	652932	160642	24.60	148859	22.80	76558	11.73	556353	85.21	513315	78.62
50_RAND_2_5	698040	97711	14.00	92329	13.23	73014	10.46	513017	73.49	481755	69.02
50_CLUS_2_1	610544	112096	18.36	139152	22.79	78077	12.79	506119	82.90	408620	66.93
50_CLUS_2_2	578322	139556	24.13	119392	20.64	239085	41.34	564390	97.59	497730	86.06
50_CLUS_2_3	530415	173395	32.69	156094	29.43	143525	27.06	613740	115.71	507452	95.67
50_CLUS_2_4	652932	153970	23.58	143029	21.91	116558	17.85	618327	94.70	482252	73.86
50_CLUS_2_5	698040	102501	14.68	92950	13.32	43730	6.26	408389	58.51	423140	60.62
50_CPCD_2_1	610544	177268	29.03	128684	21.08	17316	2.84	478348	78.35	372562	61.02
50_CPCD_2_2	578322	199029	34.41	180104	31.14	70191	12.14	524608	90.71	480088	83.01
50_CPCD_2_3	530415	125010	23.57	126768	23.90	55186	10.40	467953	88.22	389934	73.51
50_CPCD_2_4	652932	139720	21.40	135239	20.71	49700	7.61	429194	65.73	305194	46.74
50_CPCD_2_5	698040	147936	21.19	122624	17.57	18843	2.70	416731	59.70	418570	59.96
50_RAND_3_1	611295	110497	18.08	125101	20.46	137928	22.56	446495	73.04	444027	72.64
50_RAND_3_2	579037	144910	25.03	165743	28.62	137595	23.76	610555	105.44	526875	90.99
50_RAND_3_3	531236	211480	39.81	190871	35.93	87735	16.52	652890	122.90	595919	112.18
50_RAND_3_4	654049	204656	31.29	145960	22.32	19406	2.97	577585	88.31	520292	79.55
50_RAND_3_5	699080	166566	23.83	141173	20.19	43075	6.16	489669	70.04	474815	67.92
50_CLUS_3_1	611295	121869	19.94	137068	22.42	102956	16.84	539522	88.26	454680	74.38
50_CLUS_3_2	579037	177949	30.73	203260	35.10	75624	13.06	496139	85.68	456317	78.81
50_CLUS_3_3	531236	188946	35.57	188059	35.40	65475	12.33	565916	106.53	524334	98.70
50_CLUS_3_4	654049	112613	17.22	79288	12.12	69706	10.66	585684	89.55	544368	83.23
50_CLUS_3_5	699080	108955	15.59	100832	14.42	69475	9.94	507394	72.58	454337	64.99
50_CPCD_3_1	611295	153613	25.13	154136	25.21	54556	8.92	505532	82.70	409733	67.03
50_CPCD_3_2	579037	198067	34.21	216721	37.43	35240	6.09	632220	109.18	481042	83.08
50_CPCD_3_3	531236	142718	26.87	140646	26.48	64596	12.16	477960	89.97	346648	65.25
50_CPCD_3_4	654049	135683	20.75	141923	21.70	28559	4.37	425120	65.00	365988	55.96
50_CPCD_3_5	699080	152797	21.86	117240	16.77	35722	5.11	427656	61.17	402089	57.52
75_RAND_2_1	772435	210751	27.28	178496	23.11	118318	15.32	821395	106.34	780404	101.03
75_RAND_2_2	780361	203558	26.09	164115	21.03	140567	18.01	858513	110.01	714640	91.58
75_RAND_2_3	808203	223183	27.61	233986	28.95	95318	11.79	867287	107.31	801126	99.12
75_CLUS_2_1	772435	175378	22.70	158096	20.47	146760	19.00	960092	124.29	744532	96.39
75_CLUS_2_2	780361	192213	24.63	181559	23.27	127157	16.29	811136	103.94	645325	82.70
75_CLUS_2_3	808203	216559	26.80	182690	22.60	121210	15.00	987165	122.14	799485	98.92
75_CPCD_2_1	772435	198279	25.67	193313	25.03	68253	8.84	767161	99.32	631867	81.80
75_CPCD_2_2	780361	158445	20.30	177454	22.74	77988	9.99	701347	89.87	647925	83.03
75_CPCD_2_3	808203	174004	21.53	180719	22.36	59467	7.36	783842	96.99	700438	86.67
75_RAND_3_1	774140	176473	22.80	164639	21.27	112401	14.52	878017	113.42	753141	97.29
75_RAND_3_2	782381	259844	33.21	234453	29.97	128953	16.48	903349	115.46	824284	105.36
75_RAND_3_3	810106	254426	31.41	235819	29.11	99310	12.26	915247	112.98	809916	99.98
75_CLUS_3_1	774140	169253	21.86	174188	22.50	127175	16.43	909365	117.47	768385	99.26
75_CLUS_3_2	782381	209017	26.72	203689	26.03	136007	17.38	840036	107.37	766205	97.93
75_CLUS_3_3	810106	205313	25.34	237972	29.38	66059	8.15	1,006,955	124.30	871181	107.54
75_CPCD_3_1	774140	248491	32.10	227597	29.40	87368	11.29	839908	108.50	690581	89.21
75_CPCD_3_2	782381	267363	34.17	265708	33.96	103766	13.26	788126	100.73	657075	83.98
75_CPCD_3_3	810106	257667	31.81	207073	25.56	41894	5.17	781260	96.44	639111	78.89
100_RAND_2_1	1,072,407	332020	30.96	277317	25.86	122109	11.39	1,275,821	118.97	1,169,467	109.05
100_CLUS_2_1	1,072,407	266633	24.86	277927	25.92	216976	20.23	1,285,608	119.88	1,099,816	102.56
100_CPCD_2_1	1,072,407	285790	26.65	187515	17.49	199586	18.61	1,199,016	111.81	899469	83.87
100_RAND_3_1	1,074,809	364044	33.87	405311	37.71	102135	9.50	1,313,254	122.18	1,074,617	99.98
100_CLUS_3_1	1,074,809	334929	31.16	290559	27.03	189203	17.60	1,301,508	121.09	1,073,135	99.84
100_CPCD_3_1	1,074,809	298355	27.76	190291	17.70	156586	14.57	955683	88.92	866282	80.60
Average			**25.97**		**24.13**		**13.13**		**97.27**		**84.24**

Table 18
Categories of reloading effort for 3L-PDP variants 1A, 1B and 2 (complete results).

Instance	Variant 2 Reloading quantity in %			Variant 1A Reloading quantity in %				Variant 1B Reloading quantity in %			
	Total	Repos	Rotate	Total	Block	Repos	Rotate	Total	Block	Repos	Rotate
50_RAND_2_1	13.99	12.52	1.47	81.92	56.61	23.81	1.51	77.14	48.78	25.13	3.23
50_RAND_2_2	16.78	16.55	0.24	107.66	67.27	38.56	1.83	99.74	59.93	38.35	1.46
50_RAND_2_3	15.39	13.50	1.89	121.96	76.67	44.60	0.70	108.39	69.30	38.28	0.81
50_RAND_2_4	11.73	9.17	2.56	85.21	61.42	22.25	1.53	78.62	48.67	28.49	1.46
50_RAND_2_5	10.46	8.19	2.27	73.49	47.06	24.54	1.90	69.02	43.67	23.41	1.94
50_CLUS_2_1	12.79	12.79	0.00	82.90	50.20	30.93	1.77	66.93	42.55	22.95	1.43
50_CLUS_2_2	41.34	40.40	0.94	97.59	71.65	24.61	1.33	86.06	54.49	30.44	1.13
50_CLUS_2_3	27.06	22.75	4.31	115.71	76.67	38.60	0.44	95.67	51.30	40.39	3.98
50_CLUS_2_4	17.85	14.22	3.63	94.70	69.98	24.04	0.68	73.86	45.98	26.70	1.18
50_CLUS_2_5	6.26	4.73	1.53	58.51	37.49	20.33	0.68	60.62	36.59	22.08	1.95
50_CPCD_2_1	2.84	2.47	0.36	78.35	51.19	26.24	0.92	61.02	41.79	18.44	0.79
50_CPCD_2_2	12.14	11.36	0.78	90.71	60.81	28.23	1.68	83.01	52.34	29.45	1.22
50_CPCD_2_3	10.40	8.25	2.15	88.22	56.39	31.03	0.80	73.51	42.33	29.56	1.63
50_CPCD_2_4	7.61	4.77	2.84	65.73	47.32	17.46	0.95	46.74	25.62	19.80	1.33
50_CPCD_2_5	2.70	2.46	0.24	59.70	37.58	21.71	0.41	59.96	36.88	22.17	0.92
50_RAND_3_1	22.56	19.09	3.47	73.04	49.05	22.10	1.89	72.64	44.17	26.09	2.38
50_RAND_3_2	23.76	22.86	0.90	105.44	76.21	28.67	0.56	90.99	66.64	23.75	0.59
50_RAND_3_3	16.52	15.95	0.56	122.90	100.59	21.13	1.18	112.18	74.96	35.58	1.64
50_RAND_3_4	2.97	2.97	0.00	88.31	62.97	24.80	0.55	79.55	58.28	21.07	0.20
50_RAND_3_5	6.16	5.26	0.90	70.04	46.70	23.22	0.12	67.92	51.35	16.31	0.26
50_CLUS_3_1	16.84	16.60	0.24	88.26	58.70	27.84	1.72	74.38	49.70	22.56	2.12
50_CLUS_3_2	13.06	11.27	1.79	85.68	67.76	17.00	0.92	78.81	58.52	19.29	1.00
50_CLUS_3_3	12.33	12.33	0.00	106.53	78.68	26.13	1.73	98.70	65.52	30.67	2.51
50_CLUS_3_4	10.66	10.01	0.65	89.55	59.03	28.81	1.70	83.23	50.94	31.81	0.49
50_CLUS_3_5	9.94	9.38	0.56	72.58	49.24	22.00	1.35	64.99	41.92	19.07	4.00
50_CPCD_3_1	8.92	7.72	1.20	82.70	60.81	19.66	2.22	67.03	46.97	18.09	1.97
50_CPCD_3_2	6.09	5.96	0.13	109.18	78.23	30.19	0.77	83.08	55.95	26.05	1.08
50_CPCD_3_3	12.16	11.12	1.04	89.97	65.48	23.75	0.74	65.25	42.64	21.64	0.98
50_CPCD_3_4	4.37	3.51	0.86	65.00	46.74	17.89	0.37	55.96	36.18	18.38	1.40
50_CPCD_3_5	5.11	4.25	0.86	61.17	43.33	16.91	0.93	57.52	39.02	17.66	0.84
75_RAND_2_1	15.32	14.64	0.68	106.34	78.21	26.38	1.75	101.03	63.99	34.52	2.51
75_RAND_2_2	18.01	17.26	0.75	110.01	75.06	32.72	2.23	91.58	59.61	30.32	1.65
75_RAND_2_3	11.79	10.97	0.83	107.31	72.60	32.82	1.89	99.12	62.15	35.15	1.83
75_CLUS_2_1	19.00	15.87	3.13	124.29	79.66	41.21	3.43	96.39	58.04	35.83	2.51
75_CLUS_2_2	16.29	15.55	0.75	103.94	70.91	30.99	2.05	82.70	57.45	23.25	2.00
75_CLUS_2_3	15.00	12.03	2.97	122.14	82.79	38.24	1.12	98.92	64.14	32.84	1.94
75_CPCD_2_1	8.84	7.05	1.79	99.32	66.95	30.16	2.20	81.80	48.41	29.29	4.10
75_CPCD_2_2	9.99	9.24	0.76	89.87	59.62	26.24	4.02	83.03	46.22	32.63	4.18
75_CPCD_2_3	7.36	5.69	1.67	96.99	64.32	29.64	3.03	86.67	47.02	37.20	2.45
75_RAND_3_1	14.52	13.88	0.64	113.42	76.35	36.27	0.81	97.29	60.81	34.86	1.62
75_RAND_3_2	16.48	12.48	4.00	115.46	91.61	22.20	1.66	105.36	78.67	24.23	2.46
75_RAND_3_3	12.26	11.55	0.71	112.98	82.85	29.94	0.19	99.98	67.45	31.83	0.70
75_CLUS_3_1	16.43	15.22	1.21	117.47	81.38	34.56	1.53	99.26	67.11	31.14	1.00
75_CLUS_3_2	17.38	16.51	0.88	107.37	86.52	19.30	1.55	97.93	62.11	33.73	2.09
75_CLUS_3_3	8.15	8.15	0.00	124.30	86.54	36.42	1.33	107.54	76.16	29.97	1.41
75_CPCD_3_1	11.29	11.02	0.27	108.50	77.55	28.99	1.95	89.21	57.42	31.05	0.74
75_CPCD_3_2	13.26	12.02	1.25	100.73	73.40	26.50	0.83	83.98	59.54	23.14	1.30
75_CPCD_3_3	5.17	4.44	0.73	96.44	66.95	27.80	1.69	78.89	48.27	29.10	1.52
100_RAND_2_1	11.39	10.20	1.18	118.97	79.52	37.93	1.52	109.05	65.93	40.03	3.09
100_CLUS_2_1	20.23	18.22	2.01	119.88	83.64	33.86	2.38	102.56	63.87	36.21	2.48
100_CPCD_2_1	18.61	16.29	2.32	111.81	68.25	41.20	2.36	83.87	46.67	35.19	2.02
100_RAND_3_1	9.50	7.88	1.63	122.18	95.48	24.99	1.72	99.98	69.74	27.91	2.34
100_CLUS_3_1	17.60	16.02	1.58	121.09	89.37	30.30	1.42	99.84	65.94	32.25	1.66
100_CPCD_3_1	14.57	12.75	1.82	88.92	63.05	24.51	1.35	80.60	51.65	26.18	2.77
Average	**13.13**	**11.80**	**1.33**	**97.27**	**67.86**	**27.97**	**1.44**	**84.24**	**54.28**	**28.18**	**1.78**

Table 19
Maximal loaded volumes for different 3L-PDP variants (complete results).

Instance	Variant 5		Variant 4*		Variant 3*		Variant 2		Variant 1A		Variant 1B	
	Volume	%	Volume	%	Volume	%	Volume	%	Volume	%	Volume	%
50_RAND_2_1	26657	59.24	25143	55.87	26537	58.97	27440	60.98	30979	68.84	28860	64.13
50_RAND_2_2	25996	57.77	26047	57.88	25244	56.10	27117	60.26	27118	60.26	29190	64.87
50_RAND_2_3	24245	53.88	24272	53.94	25118	55.82	27058	60.13	28961	64.36	27810	61.80
50_RAND_2_4	24929	55.40	21783	48.41	24808	55.13	23216	51.59	26649	59.22	26562	59.03
50_RAND_2_5	27212	60.47	24735	54.97	26107	58.02	27513	61.14	29990	66.64	28864	64.14
50_CLUS_2_1	31101	69.11	31855	70.79	30362	67.47	27527	61.17	27494	61.10	31142	69.20
50_CLUS_2_2	25830	57.40	25274	56.16	24916	55.37	26054	57.90	27319	60.71	27714	61.59
50_CLUS_2_3	25204	56.01	25204	56.01	27156	60.35	26874	59.72	28229	62.73	28952	64.34
50_CLUS_2_4	26350	58.56	25761	57.25	24824	55.16	25677	57.06	27458	61.02	27996	62.21
50_CLUS_2_5	28805	64.01	28805	64.01	29381	65.29	29718	66.04	29898	66.44	29747	66.11
50_CPCD_2_1	29543	65.65	28965	64.37	29637	65.86	29941	66.54	31724	70.50	30457	67.68
50_CPCD_2_2	29182	64.85	29182	64.85	29155	64.79	29294	65.10	29826	66.28	30485	67.74
50_CPCD_2_3	27536	61.19	27705	61.57	27759	61.69	27818	61.82	29876	66.39	29758	66.13
50_CPCD_2_4	27247	60.55	28820	64.04	25733	57.18	28879	64.18	28977	64.39	28892	64.20
50_CPCD_2_5	30730	68.29	30286	67.30	30088	66.86	29944	66.54	30516	67.81	30812	68.47
50_RAND_3_1	28707	63.79	26603	59.12	27357	60.79	27308	60.68	28137	62.53	28681	63.74
50_RAND_3_2	25991	57.76	25307	56.24	25886	57.52	27979	62.18	30287	67.30	29718	66.04
50_RAND_3_3	25621	56.94	26299	58.44	25602	56.89	27236	60.52	29162	64.80	29038	64.53
50_RAND_3_4	24914	55.36	24834	55.19	24936	55.41	25021	55.60	25036	55.64	25025	55.61
50_RAND_3_5	28150	62.56	26564	59.03	26371	58.60	27916	62.04	26847	59.66	27329	60.73
50_CLUS_3_1	28234	62.74	26779	59.51	28033	62.30	28250	62.78	28744	63.88	28779	63.95
50_CLUS_3_2	24172	53.71	25220	56.05	25311	56.25	27102	60.23	28549	63.44	27350	60.78
50_CLUS_3_3	24338	54.09	24212	53.80	24224	53.83	24435	54.30	28334	62.96	27083	60.18
50_CLUS_3_4	24806	55.12	24806	55.12	25638	56.97	24907	55.35	25755	57.23	26494	58.88
50_CLUS_3_5	29504	65.56	29983	66.63	29744	66.10	32116	71.37	30600	68.00	30724	68.28
50_CPCD_3_1	30281	67.29	28850	64.11	29471	65.49	30298	67.33	29739	66.09	29760	66.13
50_CPCD_3_2	29225	64.94	28629	63.62	28611	63.58	28731	63.85	28568	63.49	28087	62.42
50_CPCD_3_3	27817	61.82	27775	61.72	27570	61.27	27819	61.82	27885	61.97	28419	63.15
50_CPCD_3_4	27306	60.68	30656	68.12	27387	60.86	28970	64.38	28963	64.36	29123	64.72
50_CPCD_3_5	29866	66.37	30110	66.91	29897	66.44	29841	66.31	30482	67.74	31168	69.26
75_RAND_2_1	27740	61.64	27747	61.66	28414	63.14	28299	62.89	29816	66.26	29114	64.70
75_RAND_2_2	28035	62.30	26620	59.16	25825	57.39	28021	62.27	28432	63.18	27961	62.14
75_RAND_2_3	27506	61.13	27143	60.32	27683	61.52	28909	64.24	28767	63.93	29957	66.57
75_CLUS_2_1	27832	61.85	28361	63.02	27739	61.64	27185	60.41	29826	66.28	30479	67.73
75_CLUS_2_2	30952	68.78	29739	66.09	28528	63.40	31016	68.93	30480	67.73	29832	66.29
75_CLUS_2_3	27964	62.14	26548	59.00	28459	63.24	27940	62.09	30566	67.92	28553	63.45
75_CPCD_2_1	27761	61.69	28870	64.15	28334	62.96	29098	64.66	30358	67.46	30421	67.60
75_CPCD_2_2	30151	67.00	28003	62.23	29405	65.34	29609	65.80	29838	66.31	28624	63.61
75_CPCD_2_3	29870	66.38	28730	63.84	28669	63.71	29292	65.09	28695	63.77	30507	67.79
75_RAND_3_1	25700	57.11	25602	56.89	23763	52.81	26309	58.46	28456	63.24	27840	61.87
75_RAND_3_2	27759	61.69	25739	57.20	25062	55.69	27829	61.84	29197	64.88	28545	63.43
75_RAND_3_3	28552	63.45	27761	61.69	26381	58.62	27655	61.46	27948	62.11	27154	60.34
75_CLUS_3_1	27882	61.96	27882	61.96	26488	58.86	27811	61.80	27813	61.81	27790	61.75
75_CLUS_3_2	27211	60.47	27154	60.34	26047	57.88	28176	62.61	28132	62.52	28729	63.84
75_CLUS_3_3	24573	54.61	25230	56.07	26327	58.50	26528	58.95	28104	62.45	27850	61.89
75_CPCD_3_1	27843	61.87	27170	60.38	28430	63.18	27748	61.66	27897	61.99	30408	67.57
75_CPCD_3_2	28240	62.76	28032	62.29	27836	61.86	28115	62.48	30456	67.68	29180	64.84
75_CPCD_3_3	28504	63.34	27917	62.04	27856	61.90	29317	65.15	29333	65.19	29919	66.49
100_RAND_2_1	27759	61.69	27996	62.21	27160	60.36	28997	64.44	29828	66.28	28472	63.27
100_CLUS_2_1	31157	69.24	27745	61.66	27712	61.58	29865	66.37	29864	66.36	31206	69.35
100_CPCD_2_1	31101	69.11	30356	67.46	29602	65.78	30307	67.35	29741	66.09	30309	67.35
100_RAND_3_1	29233	64.96	26428	58.73	25816	57.37	27798	61.77	29635	65.86	28920	64.27
100_CLUS_3_1	28683	63.74	27729	61.62	27704	61.56	29229	64.95	28589	63.53	28722	63.83
100_CPCD_3_1	29084	64.63	27713	61.58	28987	64.42	28581	63.51	29187	64.86	28554	63.45
Average		**61.75**		**60.60**		**60.54**		**62.37**		**64.32**		**64.32**

Table 20
Total iteration numbers and computing times to find the best solution (complete results).

Instance	CPU	Variant 5		Variant 4*		Variant 3*		Variant 2		Variant 1A		Variant 1B	
	seconds	Total iterations	Runtime to best in %	Total iterations	Runtime to best in %	Total iterations	Runtime to best in %	Total iterations	Runtime to best in %	Total iterations	Runtime to best in %	Total iterations	Runtime to best in %
50_RAND_2_1	300	202391	44.75	52109	75.65	53446	68.25	123489	31.77	15554	53.23	14225	56.41
50_RAND_2_2	300	126885	73.05	24110	70.60	28957	75.90	64462	48.26	8119	78.61	5024	71.14
50_RAND_2_3	300	151494	44.81	43847	71.12	35691	70.79	78827	38.25	6666	76.96	6031	65.65
50_RAND_2_4	300	151792	46.45	28498	87.66	27930	69.77	85268	36.81	11596	68.26	6421	66.24
50_RAND_2_5	300	266752	36.15	98324	63.82	92455	61.27	143485	40.17	31876	72.02	25170	46.90
50_CLUS_2_1	300	250170	22.13	84277	62.89	89275	72.73	123943	20.30	9136	75.24	10449	81.29
50_CLUS_2_2	300	123490	27.96	12600	77.37	11138	95.50	48471	48.45	4882	71.57	3523	74.28
50_CLUS_2_3	300	207934	33.51	59885	80.81	46344	79.08	71768	55.26	4983	70.58	3500	75.67
50_CLUS_2_4	300	197856	45.04	34758	72.25	30585	80.63	90610	53.22	9724	75.74	4656	74.40
50_CLUS_2_5	300	298884	41.24	104983	56.65	109427	86.22	167905	69.80	39465	52.57	27857	75.29
50_CPCD_2_1	300	61742	57.03	6131	70.50	6142	73.87	28720	62.44	2773	68.29	2369	78.53
50_CPCD_2_2	300	105669	59.22	7536	68.71	7174	77.84	30180	62.42	2583	70.55	1830	74.42
50_CPCD_2_3	300	52951	50.24	4947	81.76	4597	80.45	18384	83.53	2825	73.30	2093	83.21
50_CPCD_2_4	300	90880	72.48	6484	79.17	5881	70.91	24082	81.92	1995	66.31	1587	76.11
50_CPCD_2_5	300	139220	59.65	17216	82.07	19777	71.54	54964	56.90	5939	74.33	4510	71.15
50_RAND_3_1	600	159771	40.97	13600	85.53	10091	81.91	133464	41.25	8048	79.55	7013	77.30
50_RAND_3_2	600	23056	66.45	2797	84.44	2650	89.89	16463	57.62	2203	69.89	881	81.79
50_RAND_3_3	600	68423	39.91	4004	77.04	4246	82.97	45665	63.82	2693	79.31	1195	86.26
50_RAND_3_4	600	153389	51.81	13223	70.15	11453	76.97	120345	45.17	6169	84.98	3739	86.58
50_RAND_3_5	600	240527	55.98	26550	66.96	23683	67.57	135927	47.01	14020	65.57	9667	70.74
50_CLUS_3_1	600	284412	43.66	12799	78.76	13476	74.84	155701	48.24	4201	93.61	2497	77.34
50_CLUS_3_2	600	109358	61.55	5776	95.72	5226	85.41	94300	38.68	3605	87.01	2028	71.60
50_CLUS_3_3	600	149572	62.92	5933	78.80	5531	78.52	74043	64.12	1995	71.33	1253	88.86
50_CLUS_3_4	600	222735	41.73	10459	75.76	12813	73.40	114954	55.34	4159	82.40	2448	79.38
50_CLUS_3_5	600	304166	39.87	14655	87.35	14891	88.94	162529	47.02	5774	89.49	3896	84.26
50_CPCD_3_1	600	3152	75.62	903	78.60	881	80.87	1682	78.50	433	84.91	469	68.18
50_CPCD_3_2	600	16177	92.32	1566	83.90	1392	81.95	3570	83.64	670	82.25	530	82.71
50_CPCD_3_3	600	3170	88.26	1049	67.66	1044	85.99	2768	82.30	590	81.50	485	80.42
50_CPCD_3_4	600	23513	81.84	2658	72.14	2212	84.32	7241	81.94	1085	78.86	806	82.40
50_CPCD_3_5	600	17023	91.22	1874	70.52	1794	80.95	7328	76.43	1185	75.80	1158	84.47
75_RAND_2_1	600	126043	50.61	19399	70.19	21540	85.59	74259	42.34	4569	78.96	2847	84.95
75_RAND_2_2	600	168726	38.76	33108	73.86	35404	78.89	58942	59.62	5193	73.93	3664	78.36
75_RAND_2_3	600	102339	52.30	12970	73.52	17108	84.82	46773	51.00	4421	85.48	3246	81.24
75_CLUS_2_1	600	148068	46.16	25110	79.34	24030	84.63	42455	59.72	4318	68.90	3137	79.00
75_CLUS_2_2	600	147484	63.69	23055	96.38	22889	86.42	56445	65.65	4031	85.56	2830	80.95
75_CLUS_2_3	600	119737	54.67	11814	80.32	11356	86.58	34528	71.45	2739	91.10	2210	80.13
75_CPCD_2_1	600	47698	67.64	5059	76.33	5135	87.55	18329	85.88	2535	83.63	2300	79.64
75_CPCD_2_2	600	91039	69.01	11891	86.88	12574	77.22	34917	72.71	3933	78.55	3046	76.08
75_CPCD_2_3	600	51801	67.58	9155	82.54	8504	87.48	19852	91.18	2567	85.83	2312	74.30
75_RAND_3_1	1200	56230	68.12	4686	74.48	4547	84.53	50860	68.22	1770	87.28	2065	86.67
75_RAND_3_2	1200	49749	71.03	4120	82.27	4574	78.57	27311	68.83	1421	84.30	1346	69.11
75_RAND_3_3	1200	27153	82.58	3407	78.00	3118	83.11	19221	69.48	1766	82.81	1709	81.44
75_CLUS_3_1	1200	92624	69.25	4024	77.53	3664	70.65	21573	84.34	1432	91.35	1218	84.83
75_CLUS_3_2	1200	31360	78.80	2896	78.91	2456	92.96	7372	81.52	1332	92.19	1120	92.42
75_CLUS_3_3	1200	84338	72.83	3516	72.22	2928	74.13	10081	67.74	1398	83.90	1196	88.98
75_CPCD_3_1	1200	6595	85.01	1969	66.04	1904	82.92	4438	84.89	957	82.57	953	89.37
75_CPCD_3_2	1200	10766	91.74	1961	62.66	1798	84.88	4247	78.21	891	81.30	871	76.63
75_CPCD_3_3	1200	26569	82.51	3959	76.59	3791	79.14	11814	89.98	1253	83.71	1217	71.03
100_RAND_2_1	1200	81845	60.78	9222	86.11	9639	85.23	37656	74.47	3271	77.07	3018	86.51
100_CLUS_2_1	1200	94320	71.72	14982	83.06	14242	78.51	49142	62.38	2804	79.36	2112	73.07
100_CPCD_2_1	1200	6599	84.08	2205	81.17	2154	68.84	4138	80.63	1369	77.42	1133	69.73
100_RAND_3_1	2400	28326	89.91	3382	83.47	3262	76.17	11727	89.93	1459	85.00	1392	88.66
100_CLUS_3_1	2400	32224	86.26	3247	77.53	3136	89.88	10669	86.85	1234	70.12	1012	76.92
100_CPCD_3_1	2400	3430	89.43	1450	72.91	1379	79.19	2636	82.81	753	84.58	719	88.66
Average			**61.89**		**76.79**		**79.95**		**64.27**		**78.31**		**77.99**

References

Bischoff, E. E., & Ratcliff, M. S. W. (1995). Issues in the development of approaches to container loading. *Omega, 23*, 377–390.

Bortfeldt, A. (2012). A hybrid algorithm for the capacitated vehicle routing problem with three-dimensional loading constraints. *Computers & Operations Research, 39*, 2248–2257.

Bortfeldt, A., & Wäscher, G. (2013). Constraints in container loading – A state-of-the-art review. *European Journal of Operational Research, 229*, 1–20.

Bortfeldt, A., Hahn, T., Männel, D., & Mönch, L. (2015). Hybrid algorithms for the vehicle routing problem with clustered backhauls and 3D loading constraints. *European Journal of Operational Research, 243*, 82–96.

Crainic, T. G., Perboli, G., & Tadei, R. (2008). Extreme point-based heuristics for three-dimensional bin packing. *INFORMS Journal on Computing, 20*, 368–384.

Gendreau, M., Iori, M., Laporte, G., & Martello, S. (2006). A Tabu search algorithm for a routing and container loading problem. *Transportation Science, 40*, 342–350.

Männel, D., & Bortfeldt, A. (2016). A hybrid algorithm for the vehicle routing problem with pickup and delivery and three-dimensional loading constraints. *European Journal of Operational Research, 254*, 840–858.

Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2008). A survey on pickup and delivery problems. Part II: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft, 58*, 81–117.

Pollaris, H., Braekers, K., Caris, A., Janssens, G., & Limbourg, S. (2015). Vehicle routing problems with loading constraints: State-of-the-art and future directions. *OR Spectrum, 37*, 297–330.

Ropke, S., & Pisinger, D. (2006). An adaptive large neighborhood search for the pickup and delivery problem with time windows. *Transportation Science, 40*, 455–472.

Toth, P., & Vigo, D. (2014). *Vehicle routing: Problems, methods, and applications* (2nd ed.). Philadelphia: MOS-SIAM series on optimization.

Xu, H., Chen, Z., Rajagopal, S., & Arunapuram, S. (2003). Solving a practical pickup and delivery problem. *Transportation Science, 37*, 347–364.

Teil IV:

Hybrid Algorithms for the Vehicle Routing Problem with Pickup and Delivery and Two-Dimensional Loading Constraints

WORKING PAPER SERIES

Hybrid Algorithms for the Vehicle Routing Problem with Pickup and Delivery and Two-dimensional Loading Constraints

Dirk Männel

Working Paper No. 10/2017

Impressum (§ 5 TMG)

Herausgeber:
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Der Dekan

Verantwortlich für diese Ausgabe:
Dirk Männel
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

Hybrid Algorithms for the Vehicle Routing Problem with Pickup and Delivery and Two-dimensional Loading Constraints

Dirk Männel
Otto von Guericke University, Universitätsplatz 2, 39106 Magdeburg, Germany
dirk.maennel@gmx.de
Phone: 0049 2054 8549220

Abstract

We extend the classical Pickup and Delivery Problem (PDP) to an integrated routing and two-dimensional loading problem, called PDP with two-dimensional loading constraints (2L-PDP). A set of routes of minimum total length has to be determined such that each request is transported from a loading site to the corresponding unloading site. Each request consists of a given set of 2D rectangular items with a certain weight. The vehicles have a weight capacity and a rectangular two-dimensional loading area. All loading and unloading operations must be done exclusively by movements parallel to the longitudinal axis of the loading area of a vehicle and without moving items of other requests. Furthermore, each item must not be moved after loading and before unloading.

The problem is of interest for the transport of rectangular-shaped items that cannot be stacked one on top of the other because of their weight, fragility or large dimensions. The 2L-PDP also generalizes the well-known Capacitated Vehicle Routing Problem with Two-dimensional Loading Constraints (2L-CVRP), in which the demand of each customer is to be transported from the depot to the customer's unloading site.

This paper proposes two hybrid algorithms for solving the 2L-PDP and each one consists of a routing and a packing procedure. Within both approaches, the routing procedure modifies a well-known large neighborhood search for the one-dimensional PDP and the packing procedure uses six different constructive heuristics for packing the items. Computational experiments were carried out using 60 newly proposed 2L-PDP benchmark instances with up to 150 requests.

Key words: Transportation, vehicle routing, packing, pickup and delivery.

1 Introduction

Vehicle routing problems widely arise in transportation logistics if companies are interested in optimizing their routes. Therefore problems like the classical capacitated vehicle routing problem (CVRP) and the classical pickup and delivery problem (PDP) have been investigated in the literature for many years. However the classical modeling does not consider constraints occurring in real world settings regarding the feasibility of the loading. To ensure that calculated routes can actually be implemented, a two-dimensional (2D) or three-dimensional (3D) modeling of cargo and loading spaces is indispensable in many situations. Therefore in the last ten years a good deal of research has been done on integrated routing problems with 2D or 3D loading constraints. Several packing constraints, e.g. concerning stacking of goods, can only be considered if customer demands are viewed as sets of 3D items. At the same time, often any reloading effort should be avoided that is any temporary or permanent repositioning or rotation of items after loading and before unloading. There are different practical reasons to forbid reloading of goods during a pickup and delivery route. Absence of manpower, tight working time, lack of equipment and shortage of space at customer sites are some of them. Moreover, the goods might be fragile, extra heavy or even hazardous. A 2D modeling instead of a 3D modeling is sufficient if the goods to be transported can be considered as rectangular items that cannot be stacked due to their weight, dimensions or fragility. Such issues arise in industries where large-sized items have to be transported, e.g. furniture, mechanical components and household appliances.

In the following, we consider the pickup and delivery problem with two-dimensional loading constraints (2L-PDP). As in the classical PDP, a number of requests have to be transported from a pickup point to a delivery point by means of homogeneous vehicles. However, in the 2L-PDP the demands consist of sets of 2D items to be placed on 2D loading areas of the vehicles. All vehicles are assumed as rear-loaded, i.e. the goods are loaded and unloaded at the rear exclusively by movements in length direction of the vehicle, while moving them in width direction is not permitted in the loading or unloading operation. Moreover, we assume that any reloading of items after loading and before unloading is not allowed.

Two hybrid algorithms for solving the 2L-PDP are proposed that consist of a routing and a packing procedure. Within both approaches, the routing procedure modifies a well-known large neighborhood search for the one-dimensional PDP and the packing procedure uses six different constructive heuristics for packing the items. Computational experiments are carried out using 60 newly proposed 2L-PDP benchmark instances with up to 150 requests.

The rest of the paper is organized as follows: In Section 2, the relevant literature is reviewed, and the problem is formulated in Section 3. Two solution approaches are described in Section 4. Computational experiments are reported in Section 5. Conclusions are drawn and an outlook to further research is given in Section 6.

2 Related work

Up to now the 2L-PDP was only considered by Malapert et al. (2008). They proposed a constraint programming approach for the loading aspects of the problem but did not report any numerical results. Therefore, we will focus on recent papers on the classical PDP with paired pickup and delivery points and on VRPs with 2D and 3D loading constraints. We refer the reader to Toth and Vigo (2014) for a comprehensive survey on vehicle routing. Although this paper only covers two-dimensional loading constraints, we want to consider papers on vehicle routing problems with 3D loading constraints in the literature review, too. Recent surveys of integrated vehicle routing problems with 2D and 3D loading constraints were published by Iori and Martello (2010, 2013) and Pollaris et al. (2015).

Following the classification schema by Parragh et al. (2008), the classical PDP is characterized by paired pickup and delivery points, i.e. each request is associated with a special pickup and a special delivery point. Moreover, the PDP deals with the transportation of goods and persons. In case of passenger transportation, there are often special constraints and objectives concerning the inconvenience of passengers. This problem category is known as dial-a-ride problems. A further distinction can be made with regard to the number of available vehicles. We will consider only the multi-vehicle case, while the single-vehicle case, representing an immediate extension of the Traveling Salesman Problem (TSP), is not considered here. Furthermore, several papers deal with PDPs with multiple depots or a heterogeneous fleet where a certain request can only be served with a subset of the available vehicles (see below).

The problem formulation for the classical PDP and the PDP with time windows (PDPTW) can be found, e.g. in Parragh et al. (2008) and in Toth and Vigo (2014). Most of the published solution methods are surveyed by Berbeglia et al. (2007) and Parragh et al. (2008). As the PDP generalizes the TSP it is NP-hard. Thus most papers have proposed heuristics and especially metaheuristics for solving the PDP. For an introduction in metaheuristic approaches, we refer the reader to Gendreau and Potvin (2010).

The classical PDP with time windows and multiple vehicles was first solved by Nanry and Barnes (2000) with a reactive tabu search approach. Mostly the minimization of the number of needed vehicles is used as first optimization criterion while the minimization of the total travel distance is the second criterion. A tabu embedded simulated annealing approach has been developed by Li and Lim (2001). These authors also have introduced the widely used Li-and-Lim set of benchmark instances for the PDPTW. Pankratz (2005) proposed a grouping genetic algorithm for the PDP while Lu and Dessouky (2006) have developed an ingenious construction heuristic. Ropke and Pisinger (2006) presented an adaptive large neighborhood search algorithm for the PDPTW which covers also multiple depots and heterogeneous fleets. A two-stage hybrid algorithm was presented by Bent and van Hentenryck (2006). The first phase uses simulated annealing to decrease the number of vehicles needed. The second phase consists of a large neighborhood search algorithm in order to reduce total travel cost. Nagata and Kobayashi (2010) introduced a very effective guided ejection search algorithm to reduce the number of vehicles needed. The minimization of the total travel distance was not considered in their approach. Outstanding results, especially for larger instances, were achieved through the neighborhood search methods by Bent and van Hentenryck (2006), by Ropke and Pisinger (2006) and by Nagata and Kobayashi (2010).

In the capacitated vehicle routing problem with 2D loading constraints (2L-CVRP) the requests consist of 2D rectangular items to be transported. The vehicles have a rectangular loading area where the items must be placed without overlapping. Furthermore, some additional constraints like LIFO constraint (Last In, First Out) and orientation constraint (see Section 3) are to be taken into account. Several metaheuristic methods for solving the 2L-CVRP were published, e.g. by Gendreau et al. (2008), Fuellerer et al. (2009), Zachariadis et al. (2009), Duhamel et al. (2011) and Wei et al. (2015). Iori et al. (2007) proposed an exact solution approach for the 2L-CVRP. Extensions of the 2L-CVRP were also considered in the literature, e.g. the 2L-VRP with time windows (Khebbache-Hadji et al., 2013), the 2L-VRP with heterogeneous fleet (Leung et al., 2013), the 2L-VRP with backhauls

(Dominguez et al., 2016) and the 2L-VRP with simultaneous pickup and delivery (Zachariadis et al., 2016).

In vehicle routing problems with three dimensional loading constraints, the items are stackable 3D rectangular boxes which must be placed inside the 3D loading space of a vehicle. Additional constraints are to observe in the 3D case, e.g. the stacking constraint (non-fragile boxes must not be placed above fragile boxes) and the support constraint (at least a given percentage of the base area of a box must be supported by other boxes if the box is not placed on the floor of a loading space). The 3L-CVRP was introduced and first solved by Gendreau et al. (2006). Further papers on 3L-CVRP were published, for example, by Tarantilis et al. (2009), Fuellerer et al. (2010), Wang et al. (2010), Wisniewski et al. (2011), Bortfeldt (2012), Zhu et al. (2012), Ruan et al. (2013), Wei et al. (2014), Zhang et al. (2015) and Tao and Wang (2015). Moura and Oliveira (2009) have first introduced and solved the VRP with time windows and 3D loading constraints (3L-VRPTW). A hybrid algorithm for solving the 3L-VRPTW was published by Bortfeldt and Homberger (2013). Zachariadis et al. (2012) consider a 3L-VRP with time windows where boxes are stacked on pallets, which in turn are loaded in vehicles. The 3L-VRP with backhauls was introduced by Bortfeldt et al. (2015). The problem was solved with an algorithm including a neighbourhood search algorithm for routing and a tree search algorithm for packing boxes. The 3L-VRP with pickup and delivery was solved with a similar algorithm by Männel and Bortfeldt (2016, 2017). The 3L-VRP with pickup and deliveries was already considered by Bartók and Imreh (2011) in a simpler fashion. These authors neglected the LIFO constraint and did not provide any numerical results.

3 Problem definition

Now the 2L-PDP is described more formally. There are given n requests each consisting of a pickup point i, a delivery point $n+i$ and a set I_i of goods that are to be transported from i to $n+i$ ($i = 1,\dots,n$). There are v_{max} identical vehicles, originally located at the single depot (denoted by 0), with a rectangular loading area with length L and width W and maximum weight capacity D. Let $V = \{0,1,\dots,n,n+1,\dots,2n\}$ be the set of all nodes, i.e. pickup and delivery points including the depot. Let E be a set of undirected edges (i,j) that connect all node pairs $(0 \le i, j \le 2n,\ i \neq j)$ and let $G = (V, E)$ be the resulting graph. Let travel costs c_{ij} $(c_{ij} \ge 0)$ be assigned to each edge (i,j) and let the travel costs be symmetric, i.e. $c_{ij} = c_{ji}$ $(0 \le i, j \le 2n,\ i \neq j)$. The sets I_i include m_i rectangular items I_{ik} and item I_{ik} has the length l_{ik} and the width w_{ik} ($i = 1,\dots,n$, $k = 1,\dots,m_i$). m is the sum $\sum m_i$ ($i = 1,\dots,n$) and denotes the total number of items.

The loading area of each vehicle is embedded in the first quadrant of a Cartesian coordinate system in such a way that the length and width of the loading area lie parallel to the x and y axis, respectively. The placement of item I_{ik} in a loading area is given by the coordinates x_{ik} and y_{ik} of the corner of the item closest to the origin of the coordinate system; in addition, a binary variable o_{ik} indicates which of the possible orientations of item I_{ik} is selected ($i = 1,...,n$, $k = 1,...,m_i$). $o_{ik}=0$ means that the item is placed with its length parallel to the x-axis, while o_{ik}=1 indicates that the item is rotated by 90° and its length is parallel to the y-axis.

A packing plan P for a loading area comprises one or more placements and is regarded as feasible if the following conditions hold:
(FP1) each placed item lies completely within the loading area,
(FP2) any two items that are placed on the same truck loading area do not overlap,
(FP3) each placed items lies with its edges parallel to the edges of the loading area.
Figure 1 shows a loading area with placed items. Each vehicle is loaded and unloaded at the rear and empty at the beginning of a route.

A feasible route R is a sequence of $2p+2$ nodes ($p \geq 1$) that starts and ends at the depot. R should include the pickup and delivery points of p different (among the n given) requests and each pickup point must precede the delivery point of the same request. A solution of the 2L-PDP is a set of v sequences (R_l, $P_{l,1},\ldots,P_{l,2p_l}$), where R_l is a route and $P_{l,q}$ is a packing plan ($l = 1,\ldots,v$, $q = 1,\ldots,2p_l$, p_l denotes the number of requests of route l). $P_{l,q}$ represents the packing plan of route l after having visited its (q+1)*th* node, i.e. after some items were loaded or unloaded at the (q+1)*th* node of route l.

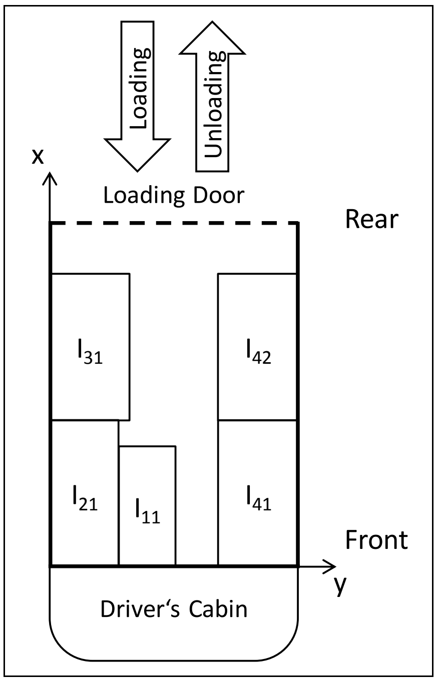


Figure 1: A loading area with placed items.

To be feasible, a solution must fulfill the following six conditions:
(F1) each route R_l starts and ends at the depot and contains at least one pickup and one delivery point ($l = 1,\ldots,v$),
(F2) each pickup point and each delivery point must occur exactly once in exactly one route,
(F3) the pickup point and the delivery point of each request lie in the same route,
(F4) each the pickup point occurs in its route before the corresponding delivery point,

(F5) all packing plans $P_{l,q}$ are feasible (l = 1,…,v, q = 1,…,2p_l), i.e. fulfill conditions (FP1) – (FP3),

(F6) the packing plan $P_{l,q}$ for a route R_l and its (q+1)*th* node contains exactly the placements for those items which are to be loaded but not (yet) to be unloaded at the first q+1 nodes of the route.

In addition, the following routing and packing constraints are to be satisfied:

(C1) LIFO *constraint for pickup points*: A packed item i of a certain request is said to be in unloading position if there is no packed item i' of another request placed between i and the rear of the vehicle. If the (q+1)*th* node of route l is a pickup point, then all items to be loaded there must be in unloading position in the packing plan $P_{l,q}$, i.e. after loading (l = 1,…,v, q = 1,…,2p_l).

(C2) LIFO *constraint for delivery points*: If the (q+1)*th* node is a delivery point, then all items to be unloaded there must be in unloading position in the packing plan $P_{l,q\text{-}1}$, i.e. before unloading (l = 1,…,v, q = 1,…,2p_l). Both LIFO constraints ensure that all items of a given request can be loaded or unloaded exclusively by movements parallel to the longitudinal axis of the loading area of a vehicle and without moving items of other requests.

(C3) *Reloading ban*: Each item I_{ik} of request i must not be moved after loading and before unloading (i = 1,…,n, k = 1,...,m_i). If the item I_{ik} is loaded at the (q+1)*th* node and unloaded at the (q'+1)*th* node of route l, its placement (x_{ik}, y_{ik}, o_{ik}) must be the same in the packing plans $P_{l,q}$, $P_{l,q+1}$,…, $P_{l,q'\text{-}1}$ (i = 1,…,n, k = 1,...,m_i, l =1,…,v, $1 \leq q < q' \leq 2p_l$).

(C4) *Weight constraint*: Each item I_{ik} has a positive weight d_{ik} (i = 1,...,n, k = 1,...,m_i) and the total weight of all items in a packing plan $P_{l,q}$ must not exceed a maximum weight capacity D (l = 1,...,v, q = 1,…,2p_l).

(C5) *Route length constraint*: The total distance of a route must not exceed a specified maximum d_{max}. This constraint can also be understood as a route duration constraint if the vehicle velocity is set to a constant.

(C6) *Route number constraint*: The number of routes v must not exceed the number of vehicles v_{max}.

Finally, the 2L-PDP consists of determining a feasible solution that meets the constraints (C1) – (C6) and minimizes the total travel distance of all routes.

The LIFO constraint for delivery points (C2) is well-known from the 2L-CVRP. At a delivery point, the LIFO constraint requires that between an item A to be unloaded and the rear of the vehicle no item B is situated that needs to be unloaded later. Otherwise item B has to be reloaded before item A can be unloaded by a pure movement in length direction. In Figure 1, the item I_{31} is in unloading position while the items I_{11} and I_{21} are not in unloading position because of the blocking item I_{31}. Both items I_{41} and I_{42} are in unloading position because they belong to the same request.

As also pickup points occur in a pickup and delivery route, a LIFO constraint to exclude reloading of goods at pickup points (C1) has to be included. At a pickup point the constraint (C1) requires that between the position of an item A just loaded and the rear of the vehicle no item B is situated that was loaded at an earlier pickup point. Again, otherwise a reloading of item B would be inevitable.

It is an essential feature of 2L-PDP that the LIFO constraints for delivery and pickup points are not sufficient to rule out any reloading effort. Furthermore, the reloading ban constraint (C3) has to be required to rule out any reloading effort. Figure 2 shows a simple example of a route and with three requests and one item per request. There exist feasible packings plans for both nodes P2 (items of request 1 and 2 loaded) and P3 (items of request 1 and 3 loaded). The packing plans fulfill both LIFO constraints (C1) and (C2), but the reloading ban (C3) is violated because box I_{11} is rotated in the second packing plan. In the example it is obviously impossible to implement the shown route without reloading box I_{11}. Hence the LIFO constraints (C1) and (C2) alone are not sufficient to rule out any reloading effort and the reloading ban constraint (C3) turns out to be necessary.

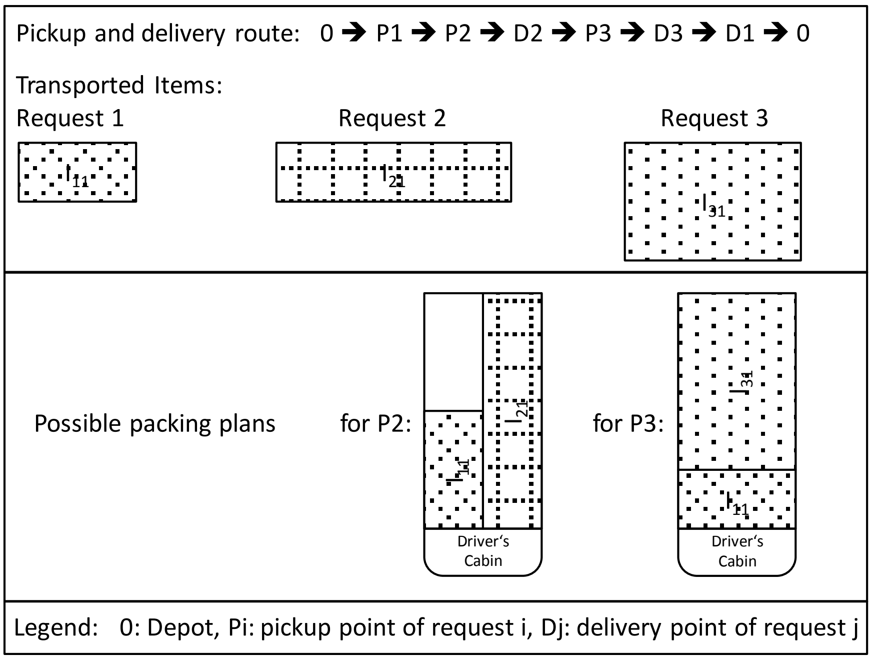

Pickup and delivery route: 0 ➔ P1 ➔ P2 ➔ D2 ➔ P3 ➔ D3 ➔ D1 ➔ 0
Transported Items:
Request 1
Request 2
Request 3
I11
I21
I31
Possible packing plans
for P2:
for P3:
Driver's Cabin
Driver's Cabin
Legend: 0: Depot, Pi: pickup point of request i, Dj: delivery point of request j

Figure 2: Packing plans with reloading for a pickup-delivery-route.

Moreover in this paper a second variant of the 2L-PDP is considered where the so-called orientation constraint (C7) is added: each placed item must lie on the loading area with its length edge parallel to the x-axis of the coordinate system (no rotation allowed). The original variant without the constraint (C7) is called in the following “Rotate” variant while the second variant is called “NoRotate” variant. In the NoRotate variant all orientation variables o_{ik} must be equal to zero in a feasible solution.

4 Two solution approaches

In this section, two solution approaches for solving the 2L-PDP are proposed. Each solution approach is a hybrid algorithm and consists of two nested procedures. The outer procedure is the routing procedure, and the packing procedure is the inner procedure. The routing procedure is basically the same for both approaches and is designed as large neighborhood search. The two approaches differ in the manner how packing checks are made and how the reloading ban constraint (C3) is taken into account.

In the first approach, the reloading ban is ensured by the routing procedure as the solution space is restricted by an additional routing constraint, the so-called independent partial route (IPR) condition. With this approach, called "Independent Partial Routes" (or IPR), it is possible to use conventional packing heuristic for packing checks. In this paper, six well-known constructive packing heuristics for the two dimensional container loading problem are integrated in a packing procedure in order to check whether a certain set of items can be packed on the loading area or not. These packing heuristics are widely used in the literature on the 2L-CVRP (see Gendreau et al., 2008, and Zachariades et al., 2009).

In the second approach ("Simultaneous Packing") the reloading ban constraint is observed by a new type of packing procedure which is able to construct a series of interrelated packing plans (see below). So the IPR condition is not needed in the second approach and this leads to a large extension of the explored solution space. Therefore, a noticeable improvement of the solution quality is expected with the Simultaneous Packing approach. However, with the larger solution space to be explored, a rising CPU-time consumption is expected, too. The main properties of both solution approaches are outlined in Table 1.

Table 1: Main properties of the two solution approaches.

Property	**Approach 1** **Independent Partial Routes**	**Approach 2** **Simultaneous Packing**
Reloading ban constraint (C3) observed by	Routing procedure	Packing procedure
Expected total travel distance	Higher	Lower
Expected CPU-power consumption	Lower	Higher

This section is organized as follows. In subsection one, the routing procedure is outlined. The second subsection presents the IPR solution approach where the 2L-PDP is solved using a straightforward packing procedure for the two-dimensional container loading problem. The implementation of the packing procedure itself is explained in subsection three. Finally, the subsection four presents the novell simultaneous packing procedure and its integration into the routing procedure.

4.1 Routing procedure

The routing procedure is derived from the (adaptive) large neighborhood search (LNS) heuristic for solving the PDP with time windows by Ropke and Pisinger (2006). In this paper a similar implementation of the routing procedure is used like in Männel and Bortfeldt (2016) before, thus the routing procedure is described in this paper only in a short fashion.

The LNS heuristics uses the „fix and optimize"-principle to construct new solutions and the neighborhood structure is defined implicitly by several removal and insert operators (heuristics). To get a new solution, first a removal operator destroys a part of the current solution, which means that some requests will be removed from their routes. Subsequently, an insert operator reinserts the removed requests at certain positions of certain routes to get a new feasible solution. The routing procedure is shown in Algorithm 1. After constructing the initial solution an iterative neighborhood search is carried out until a given time limit is exceeded. The number ξ of requests to be removed and reinserted in the solution is selected randomly within each iteration. Among the four available removal and the three available insert heuristics, one removal and one insert heuristic are selected randomly per iteration. The next solution is generated by the selected heuristics according to $s_{next} := Ih(Rh(s_{curr}, \xi))$. If s_{next} passes the acceptance test, it becomes the new current solution s_{curr}, and the best solution s_{best} is updated if s_{next} realizes a better objective function value. Otherwise, the initial solution of the next iteration s_{curr} remains unchanged. For the acceptance tests, the well-known simulated annealing rule with a geometric cooling scheme is used. The selection probabilities for the removal and insertion heuristics are fix. In the following Table 2, the available heuristics are shown.

```
2l_pdp_lns (in: problem data, parameters, out: best solution $s_{best}$)
    construct initial solution $s_{curr}$ and set $s_{best} := s_{curr}$
    while stopping criterion is not met do
        select number of requests to be removed $\xi$
        select removal heuristic $Rh$ and insertion heuristic $Ih$
        determine next solution: $s_{next} := Ih(Rh(s_{curr}, \xi))$
        check acceptance of $s_{next}$
        if $s_{next}$ is accepted then
            $s_{curr} := s_{next}$
            if $f(s_{curr}) < f(s_{best})$ then $s_{best} := s_{curr}$
    return $s_{best}$
```

Algorithm 1: LNS-based routing algorithm for the 2L-PDP.

Table 2: Removal and insertion heuristics of the LNS heuristic for 2L-PDP.

Heuristic	Description
Random removal Rh_R	Removes iteratively requests that are selected at random.
Shaw removal Rh_S	Removes iteratively requests that are related in terms of location and weight.
Worst removal Rh_W	Removes iteratively a request whose removal leads to the largest cost (total travel distance) reduction.
Tour removal Rh_T	Removes all requests from a randomly chosen route. If less than ξ requests are removed in this way, further requests will be removed with Shaw removal.
Greedy insertion Ih_G	Inserts iteratively requests into the solution such that the increase of the cost function is minimal.
Regret-2 insertion Ih_{R2}	Inserts iteratively requests into the solution such that the gap in the cost function between inserting the request into its best and its second best route is maximal.
Regret-3 insertion Ih_{R3}	Inserts iteratively requests into the solution such that the sum of two gaps in the cost function is maximal. The first gap results from inserting the request into its best and its second best route, while the second gap results from inserting the request into its best and its third best route.

4.2 IPR solution approach

In the IPR solutions approach, we want to use a conventional packing procedure for the two dimensional container loading problem to solve to the 2L-PDP. For an arbitrarily chosen route, the packing procedure has to deliver feasible packing plans for each node in the route. Each packing plan must contain exactly the items loaded on the vehicle when it is leaving the corresponding node and, furthermore, fulfill the conditions (FP1) – (FP3) and the packing related constraints (C1) – (C3) and (C7) (if necessary). In the IPR solution approach, we want to keep the packing effort low by adding a further routing constraint. This IPR constraint allows us to not to do packing checks for all nodes of a route but to restrict the packing checks to a few selected nodes within the route only. The packing plan for the other nodes can be derived from the packing plans for the selected nodes.

Definition 1: For a given node x in a 2L-PDP route, the corresponding request sequence $rs(x)$ is defined as follows. $rs(x)$ contains exactly the requests which are loaded and not yet unloaded when the vehicle is leaves the node x. The order of the requests in $rs(x)$ is given by the order of the corresponding pickup nodes within the route.

Example 1: Considering the route $0 \rightarrow P1 \rightarrow P2 \rightarrow D1 \rightarrow P3 \rightarrow P4 \rightarrow D3 \rightarrow D2 \rightarrow D4 \rightarrow 0$, the corresponding request sequences of the nodes $P4$ and $D3$ are (2, 3, 4) and (2, 4), respectively.

Definition 2:

(i) We consider a sequence $(i_1,...,i_s,i_{s+1},...,i_{2s})$ of $2s$ nodes ($s > 0$). This sequence is called “IPR block”, if its first s elements are pickup points and its last s elements are the *corresponding* delivery points and if, furthermore, the delivery points lie in inverse order of their corresponding pickup points. More formally, it should hold the following three conditions:

- $i_p \neq i_q$ for each $p, q \in (1,...,s)$ with $p \neq q$
- $1 \leq i_p \leq n$ for each $p \in (1,...,s)$
- $i_{2s-p+1} = i_p + n$ for each $p \in (1,...,s)$

(ii) A 2L-PDP route is called “IPR route” if it consists of one or more IPR blocks (plus the depot at the beginning and the end of the route).

(iii) We say that a solution of the 2L-PDP fulfills the IPR constraint if all contained routes are IPR routes.

Obviously a 2L-PDP route is an IPR route if and only if the two following conditions hold:

(1) if the vehicle visits a delivery point, then all delivery points for all items on the loading area will be visited before another pickup take place and

(2) all delivery points lie in inverse order of their corresponding pickup points, i.e. if i and j are two arbitrarily chosen requests from the route and P_i lies before P_j, then D_i must lie behind D_j in the route.

Example 2: The route $0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow P3 \rightarrow D3 \rightarrow D1 \rightarrow 0$ is not an IPR route because the vehicle does not become empty after visiting delivery point $D2$ and before visiting pickup point $P3$. The route $0 \rightarrow P3 \rightarrow P4 \rightarrow P5 \rightarrow D4 \rightarrow D5 \rightarrow D3 \rightarrow 0$ is not an IPR route because the delivery points $D4$ and $D5$ do not lie in inverse order of their corresponding pickup points. The route $0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow D1 \rightarrow P3 \rightarrow P4 \rightarrow P5 \rightarrow D5 \rightarrow D4 \rightarrow D3 \rightarrow P6 \rightarrow D6 \rightarrow 0$ is an IPR route consisting of three IPR blocks. The pickup points $P2$, $P5$ and $P6$ are called "last pickup" points because they are the last in the row of consecutive pickup points and are followed by a delivery point. Obviously, each IPR block contains exactly one last pickup point.

Definition 3: We consider an arbitrarily chosen pickup node from a 2L-PDP route with the corresponding request sequence $rs = (i_1,...,i_s)$ ($s > 0$, $1 \le i_p \le n$ for each $p \in (1,...,s)$). We say that a packing plan for the request sequence rs fulfills the *cumulative LIFO constraint* (CLC) if for each p and q with $1 \le p < q \le s$ no item of request i_p lies between an item of i_q and the rear of the vehicle, i.e. if no loading operation of an *later* loaded item of i_q is blocked by an *earlier* loaded item of i_p.

Proposition 1: Let be given an IPR route consisting of one or more IPR blocks and let exist packing plans observing conditions (FP1) – (FP3), (C7) (if necessary) and (CLC) for the request sequence of each last pickup point of each IPR block. Then feasible packing plans in terms of 2L-PDP exist for all nodes in the route which fulfill:

(i) for pickup points the conditions (FP1) – (FP3), the orientation constraint (C7) (if necessary) and the LIFO constraint for pickup points (C1),

(ii) for delivery points the conditions (FP1) – (FP3), the orientation constraint (C7) (if necessary) and the LIFO constraint for delivery points (C2) and

(iii) observe (collectively) the reloading ban constraint (C3).

Proof:

(i) In the following, the given packing plans for the last pickup points will be called "master plans". Each master plan contains placements for all requests belonging to its corresponding IPR block. Thus for each pickup node in the route, a packing plan can be derived by simply removing placements for items not yet loaded from the corresponding master plan. Obviously, the derived plans will contain the correct items and fulfill the conditions (FP1) – (FP3) together with constraint (C7) (if necessary). Because the cumulative LIFO constraint (CLC) is stronger than the LIFO constraint for pickup points (C1) all derived plans fulfill also the constraint (C1) too.

(ii) As in the proof of (i) for each delivery node, a packing plan can derived by simply removing placements for items already unloaded from the corresponding master plan. Obviously, these derived plans again will contain the correct items and fulfill the conditions (FP1) – (FP3) together with constraint (C7) (if necessary). Finally we want to prove the LIFO constraint for delivery points (C2) indirectly and assume that (C2) would not hold. In this case would exist requests *A* and *B* belonging to the same IPR block such that delivery point D_A would lie before D_B in the route and an the unloading operation of an item *a* of request *A* would be blocked by an item *b* of request *B*. Because of the structure of the IPR blocks, the pickup point P_A would lie behind the pickup point P_B, i.e. both items *a* and *b* would be also contained in the packing plan of P_A. Since all packing plans for nodes of the same IPR block are derived from one master plan, the items *a* and *b* would hold the same positions in all plans. Hence the item *b* would block the loading operation of item *a* at pickup point P_A, which would lead to a violation of constraint (C1).

(iii) If an IPR route is given, each item must only be stowed in packing plans of one IPR block. As shown in (i) and (ii), all packing plans for the other pickup and delivery points of an IPR block will be derived from the packing plan for the last pickup point by simply removing items. Hence, the positions of all items that occur in multiple packing plans remain unchanged. □

The outcome of *Proposition 1* is, that in case of IPR routes, the cumulative LIFO constraint (CLC) is sufficient that constraints (C1) – (C3) hold. Furthermore, we show in the following *Proposition 2* that constraints (C1) and (C3) are sufficient for the (CLC) constraint to hold, i.e. in case of IPR routes (CLC) and (C1) – (C3) are equivalent. So the inclusion of the (CLC) constraint neither restricts the search space additionally, nor leads to a loss of solution quality in case of the IPR solution approach.

Proposition 2: Let be given a 2L-PDP route and let packing plans exist for all pickup points in the route. Let all these plans fulfill the LIFO constraint for delivery points (C1) and the reloading ban constraint (C3). Then all these packing plans also fulfill the cumulative LIFO constraint (CLC).

Proof: We assume that the (CLC) constraint would not hold. Then would exist requests *A*, *B* and *C* (with pickup point P_A lying before P_B and P_B before P_C) and items *a* (of *A*) and *b* (of *B*) such that item *a* would lie between item *b* and the rear of the vehicle in the packing plan for pickup point P_C. Because of the reloading ban constraint (C3), these items would hold the same placements in the packing plan for the earlier pickup point P_B too, i.e. in this packing plan item *a* would also lie between item *b* and the rear of the vehicle. Thus the LIFO constraint for pickup points (C1) would be violated at pickup point P_B. □

Finally, in this section, it is to discuss how the packing procedure will be integrated into the routing procedure. In general, the LNS heuristic removes some requests (i.e. pairs of a pickup and a delivery point) of the route and reinserts some new requests. In case of the 2L-PDP, it is impossible that a route loses their "packability" by removing requests from the route. Thus, packing checks are integrated in insertion heuristics exclusively (and not in removal heuristics). The integration of the packing procedure into the insertions heuristics takes place according to two principles. First, one-dimensional checks are made before 2D packing checks are carried out. Second, all possible insertions are first evaluated and sorted by cost *before* the "expensive" packing checks are made. By this technique, called "evaluating first, packing second", the packing effort is kept low since the packing checks can be aborted each time after a few (2D-)feasible insertions have been detected. The packing checks are made “on demand”, i.e. the packing check for a certain insert possibility is not made until all other insertion possibilities for the same request and the same route with lower insertion costs have been checked in terms of packing. Whenever for a certain pair of request and route a packable insertion possibility is found, then all other insertions possibilities for this pair of request and route with higher insertion costs can be neglected from further packing checks. In this context, “packable insertion possibility” means that the route which would result from the implementation of the insert possibility is feasible in terms of packing. For more details about the integration of the packing checks into the insertion heuristics, the reader is referred to Männel and Bortfeldt (2016).

4.3 Packing procedure for IPR solution approach

In the last section, it was shown that in case of restriction to IPR routes a conventional packing procedure is sufficient to ensure the existence of feasible packing plans for the 2L-PDP. The packing procedure must be applied only to the corresponding request sequences of the last pickup points of the routes. It has to deliver packing plans which fulfill the conditions (FP1) – (FP3), the constraint (C7) (if necessary) and the cumulative LIFO constraint (CLC). As described before, then it is ensured that packing plans for all nodes satisfying the constraints (C1) – (C3) can be derived.

In this section, the packing procedure *packing_check_rs* is introduced and it is described how the procedure performs the packing check for a certain request sequence $rs = (i_1,\ldots,i_s)$. The procedure is similar to Zachariades et al. 2009. The packing procedure uses six constructive heuristics $H_1 - H_6$ and five orderings $Ord_1 - Ord_5$ for the items shown in Table 3 and 4, respectively. To observe the cumulative LIFO constraint (CLC), the items are ordered corresponding to the position of their request in the request sequence as primary criterion. The packing procedure is shown in Algorithm 2.

Table 3: Heuristics used in the packing procedure.

Heuristic	Description	Main idea for choosing the item and the allocation point
H_1	Bottom-Left Fill (Chazelle 1983)	Minimize the allocation points x-coordinate first and y-coordinate second
H_2	Left-Bottom Fill (Chazelle 1983)	Minimize the allocation points y-coordinate first and x-coordinate second
H_3	Touching Perimeter (Lodi et al. 1999)	Maximize the sum of the items common edges with other items and the loading area edges
H_4	Touching Perimeter No Walls (Lodi et al. 1999)	Maximize the sum of the items common edges with other items
H_5	Min Area heuristic (Zachariadis et al. 2009)	Minimize the size of the allocation points corresponding rectangular surface
H_6	LBFH (Lowest Reference Line Best Fit heuristic) (Leung et al. 2011)	Uses predictive strategy with changing the placing order of items, the best fitting item for the lowest rectangular space will be chosen

Table 4: Characteristics of the orderings used in the packing procedure.

Ordering	First criterion	Second criterion	Third criterion
Ord_1	position of request in *rs*	area l*w (descending)	longer side max(l,w) (descending)
Ord_2	position of request in *rs*	width w (descending)	length l (descending)
Ord_3	position of request in *rs*	length l (descending)	width w (descending)
Ord_4	position of request in *rs*	ratio of longer to shorter side max(l,w) / min(l,w) (descending)	area l*w (descending)
Ord_5	position of request in *rs*	ratio of longer to shorter side max(l,w) / min(l,w) (ascending)	area l*w (descending)

```
packing_check_rs (in: request sequence rs, out: boolean result)
    if cache.contains(rs) then                        // if rs was already checked => take result from cache
        return cache.get-result(rs)
    for u := 1 to 5 do
        is := build-item-sequence(rs, $Ord_u$)           // build item sequence for rs using ordering $Ord_u$
        for v := 1 to 6 do
            if heuristic $H_v$ can pack item sequence is then
                cache.set-result(rs, true)            // save positive result for rs in cache and return
                return true
    cache.set-result(rs, false)                       // all (u, v)-pairs were tried without success
    return false                                      // save negative result for rs in cache and return
```

Algorithm 2: Packing procedure *packing_check_rs*.

The procedure *packing_check_rs* takes a request sequence as input and returns a boolean value (true or false) indicating whether a feasible packing plan was found or not. First, the procedure checks if the request sequence is contained in the packing cache, which means that it was already checked. In this case the result is taken from the cache and the procedure terminates. Otherwise two nested loops are executed, the outer loop iterates over the orderings and the inner loop iterates over the heuristics. In each iteration of the outer loop, first the item sequence for the current ordering is built and then up to six heuristics are applied. The procedure terminates with result "true" if one heuristic can pack the item sequence *is*, otherwise the procedure returns "false" after all 30 combinations of heuristics and orderings were tried without success. In both cases, the result is saved in the packing cache before leaving the procedure. The usage of the packing cache provides a large speedup of the algorithm because the retrieval of the check result from the cache is 100 to 1000 times faster than the repetition of the packing check with the six heuristics and five orderings.

In the following, the packing heuristic H_1 (Bottom-Left Fill) is explained in detail. Central component of this heuristic is *posList*, a set of so-called allocation points (positions where the lower left corner of new items can be placed). Initially, *posList* contains only the position (0, 0). The items will be placed "item by item", respecting the item sequence which was created by the ordering in advance. Each time after an item was placed, the set *posList* will be updated, no more usable allocation points will be removed and new allocation points will be added. The new allocation points are so-called extreme points (see Crainic et al. 2008) generated by projection of the upper left corner of the placed item into –y direction and of the lower right corner into –x direction. The allocation points emerge where the projections “hit” the sides of already placed items or the loading area. Thereby, each projection can create more than one new allocation points. In Figure 3 is shown a loading area with some placed items. The allocation points are marked as bold circles.

In the Bottom-Left Fill heuristic the position for placing an item is selected from *posList* as follows. All allocations points in *posList* are checked if the considered item can be placed feasible at this allocation point, i.e. without overlapping or violating the cumulative LIFO loading constraint. Amongst all feasible allocation points, the one with the lowest x-coordinate is selected, ties are broken by the lowest y-coordinate. Thus, the Bottom-Left Fill heuristic tends to generate packing plans consisting of strips parallel to the y-axis. If for one item no feasible placement can be found, then the heuristic terminates without success, otherwise the heuristic terminates successfully when all items are placed. In case of the original 2L-PDP problem variant (Rotate), each allocation point is considered twice, one time for placing the item in original orientation and a second time for a placing it in rotated orientation, while in the second problem variant (NoRotate) only the original orientation is considered.

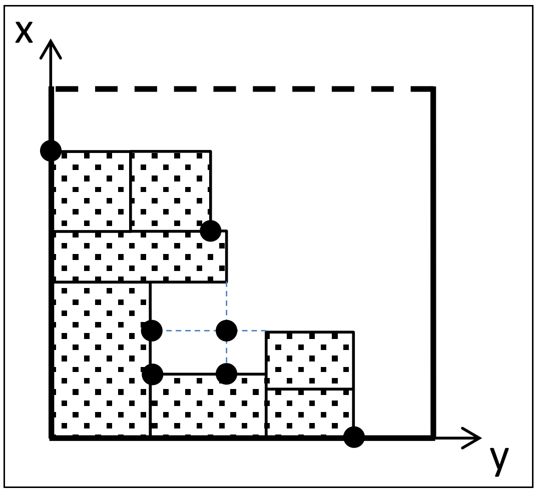


Figure 3: Allocation points marked as bold circles.

The Left-Bottom Fill heuristic H_2 works like the heuristic H_1 with the only difference, that amongst all feasible placements the allocation point with the lowest y-coordinate will be selected, ties are broken by the lowest x-coordinate. Thus, the Left-Bottom Fill heuristic tends to build packing plans made up by strips parallel to the x-axis. This approach can be useful if an extra long item must be loaded late.

In case of the Touching Perimeter heuristic H_3 for each feasible allocation point in *posList*, the total touching perimeter value of the inserted item is calculated. The touching perimeter is evaluated as the sum of the common edges of the inserted item with the edges of the already inserted items and the edges of the loading area. The item is placed at that allocation point which reaches the maximal touching perimeter value. The Touching Perimeter heuristic tends to initially place the items at the edges of the loading area and later fill the inner parts of it.

The Touching Perimeter No Walls heuristic H_4 uses the same principle like H_3, but the touching perimeter is calculated only considering common edges of the item to place with already placed items, common edges with the loading area are not taken into account. Thus, this heuristic tends to fill the inner part of the loading area earlier and cover its edges later.

In case of the Min Area heuristic H_5, the size of its corresponding rectangular surface area is calculated for each feasible allocation point. The loading position selected is the one yielding the minimum surface area. Figure 4 shows an example of a loading area with two arranged items (dotted) and three possible allocation points. The corresponding rectangular surface areas are shown as squared. The main goal of the heuristic is achieving a high degree of utilization of the vehicle's loading areas.

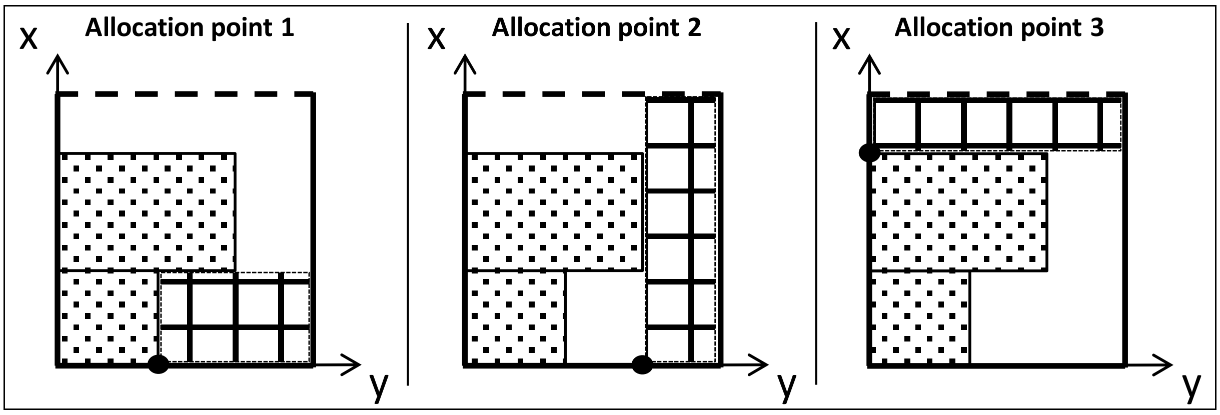


Figure 4: Example of corresponding loading areas.

The last heuristic H_6 is the LBFH heuristic (Lowest Reference Line Best Fit heuristic). The LBFH heuristic determines in each iteration first the lowest non-occupied rectangular space (with minimum x-coordinate). Then all not yet placed items of the currently processed request are tested whether they fit into the considered space. Fitting items furthermore score "fitness points" if they fill out the complete width of the considered space or if their upper edge reaches the same x-value like the adjacent items placed on the left or right of the considered space. Finally, the item with the best fitness value gets placed. Ties are broken by the items position in the sequence *is*. The main goal of the LBFH heuristic is to make the best use of the available space and reduce waste. To do so the heuristic uses a predictive strategy and changes the placing order of items belonging to the same request. For more details, the reader is referred to Leung et al. (2011).

The order of the six heuristics H_1 to H_6 is chosen so that the most simple heuristics (with smallest computational effort) Bottom-Left Fill and Left-Bottom Fill are tried first in the packing procedure *packing_check_rs*. If they fail to construct a feasible packing plan, they are followed by the more complex heuristics Touching Perimeter, Touching Perimeter No Wall, Min Area and LBFH.

4.4 Simultaneous Packing approach

In section 4.2, we introduced the IPR solution approach which heavily restricts the solution space by incorporating two additional requirements, namely (1) and (2), to the routes to satisfy the packing related constraints (C1) – (C3) and (C7) (if necessary). Now we want to introduce the Simultaneous Packing approach which drops the additional requirement (1) and allows the algorithm to explore a much larger solution space.

Definition 4:

(i) A 2L-PDP route is called "LIFO route" if the condition holds that each two delivery points in the route lie in inverse order of their corresponding pickup points. More formally, if i and j are two arbitrarily chosen requests from the route and P_i lies before P_j, then D_i must lie behind D_j in the route.

(ii) A partial route of a LIFO route is called "LIFO route block" (LRB) if the vehicle is empty when arriving at the first node and empty when leaving the last node of the partial route and if, furthermore, the vehicle does not become empty within the partial route.

(iii) We say that a solution of the 2L-PDP fulfills the LIFO route condition if all of its contained routes are LIFO routes.

Example 3: The route $0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow P3 \rightarrow P4 \rightarrow D4 \rightarrow P5 \rightarrow D5 \rightarrow D3 \rightarrow D1 \rightarrow P6 \rightarrow P7 \rightarrow D7 \rightarrow D6 \rightarrow 0$ is a LIFO route consisting of two LIFO route blocks. The first LRB starts at $P1$ and ends at $D1$, while the second LRB starts at $P6$ and ends at $D6$. The first LRB contains three last pickup points $P2$, $P4$ and $P5$ with the corresponding request sequences (1, 2), (1, 3, 4) and (1, 3, 5). The second LRB contains only one last pickup point $P7$ with the request sequence (6, 7).

In the Simultaneous Packing approach, we want to continue applying the idea of restricting the packing checks to the last pickup points within the route. The packing plans for the other nodes should be derived from the packing plans for the last pickup points. Thus it is to investigate under which additional circumstances the existence of packing plans for the last pickups in a route can ensure that feasible packing plans for all nodes of the route exist.

Proposition 3: Let be given a LIFO route consisting of one or more LIFO route blocks and let exist packing plans observing conditions (FP1) – (FP3), (C7) (if necessary) and (CLC) for the request sequences of all last pickup points of the route. Furthermore, let these packing plans fulfill the reloading ban constraint (C3), i.e. each item should hold the same placement in all plans containing this item. Then feasible packing plans in terms of 2L-PDP exist for all nodes in the route which fulfill:

(i) for pickup points the conditions (FP1) – (FP3), the orientation constraint (C7) (if necessary) and the LIFO constraint (C1),

(ii) for delivery points the conditions (FP1) – (FP3), the orientation constraint (C7) (if necessary) and the LIFO constraint (C2) and

(iii) observe (collectively) the reloading ban constraint (C3).

Proof:

(i) Each pickup point which is not a last pickup is followed by a sequence of one or more consecutive pickup points. This sequence of pickup points ends with a last pickup point which is followed by a delivery point. We call this last pickup point "corresponding last pickup point" of the original pickup point. Thus for each pickup point a packing plan can be derived by simply removing placements for items not yet loaded from the packing plan of the corresponding last pickup point. Obviously, the derived plans will contain the correct items and fulfill the conditions (FP1) – (FP3) together with constraint (C7) (if necessary). Since the cumulative LIFO constraint (CLC) is stronger than the LIFO constraint for pickup points (C1), all derived plans fulfill the constraint (C1), too.

(ii) For each delivery point in a LIFO route, the "corresponding last pickup point" is found as follows. We consider all pickup points lying in the route before the regarded delivery point and choose the last of them as corresponding last pickup point. Obviously, for each delivery point exists at least one pickup point lying in the route before it. Furthermore, the so selected pickup point is a last pickup point because it directly precedes the regarded delivery point or there are only delivery points located between them. Thus for each delivery point, a packing plan can be derived by simply removing placements for items already unloaded from the packing plan of its corresponding last pickup point. The derived packing plans will contain the correct items and fulfill the conditions (FP1) – (FP3) together with constraint (C7) (if necessary). Finally, the proof of the LIFO constraint for delivery points (C2) can take place indirectly like the proof of *Proposition* 1 because in case of LIFO routes, it is still ensured that delivery points lie in invers order of their corresponding pickup points. Thus a violation of constraint (C2) would result in a violation of constraint (C1) at a pickup point.

(iii) The packing plans for all nodes will be generated by simply removing placements from the packing plans of the last pickup points. Since these original packing plans of the last pickup points observe reloading ban constraint (C3), the derived plans will fulfill it too. □

In section 4.2, it was shown that in case of IPR routes the inclusion of the cumulative LIFO constraint (CLC) does not lead to an additional restriction of the search space. In case of LIFO routes, it holds the same. The proof is not presented here because it is almost identical to the proof of *Proposition* 2.

The outcome of *Proposition* 3 is that the concept of packing checks from the IPR approach (only performing packing checks for the request sequences of the last pickup points) can be taken over to the Simultaneous Packing approach. However, the packing procedure now must additionally ensure the reloading ban constraint for the last pickup points. This check must be made per LIFO route block because only last pickup points contained in the same LIFO route block have common requests and common items. Thus for last pickup points which do not belong to the same LRB, the reloading ban constraint is observed automatically.

Example 3 (continued): To check the LRB $P1 \rightarrow P2 \rightarrow D2 \rightarrow P3 \rightarrow P4 \rightarrow D4 \rightarrow P5 \rightarrow D5 \rightarrow D3 \rightarrow D1$ in terms of packing for the 2L-PDP the following steps must be performed:

- finding packing plans for the three request sequences (1, 2), (1, 3, 4), (1, 3, 5),
- checking if the items of request 1 hold the same placements in all three plans,
- checking if the items of request 3 hold the same placements in the second and third plan.

To check the whole LIFO route from *Example* 3 in terms of packing for the 2L-PDP, it furthermore has to be checked (independently) if a feasible packing plan for the request sequence (6, 7) exists. Since the second LRB $P6 \rightarrow P7 \rightarrow D7 \rightarrow D6$ contains only one last pickup point, there are no interdependencies between packing plans of the second LRB to observe. The packing plans for the individual points in the route can be derived as follows from the plans of the last pickup points:

- for $P1$, $D2$ from the plan of $P2$ (request sequence (1, 2)),
- for $P3$, $D4$ from the plan of P4 (request sequence (1, 3, 4)),
- for $D5$, $D3$, $D1$ from the plan of $P5$ (request sequence (1, 3, 5)),
- for $P6$, $D7$, $D6$ from the plan of $P7$ (request sequence (6, 7)).

Proposition 4: Let rs_1 and rs_2 be two request sequences with $p+q$ and $p+s$ requests, respectively (p, q, $s > 0$). Furthermore, let the first p requests of both request sequences be identically:
$rs_1 = (i_1,...,i_p,i_{p+1},...,i_{p+q})$, $rs_2 = (i_1,...,i_p,i_{p+q+1},...,i_{p+q+s})$. If v and u ($v \in \{1,...,6\}$, $u \in \{1,...,5\}$) exist such that the heuristic H_v can pack both request sequences using the ordering Ord_u, then the resulting packing plans for both request sequences fulfill the reloading ban constraint, i.e. the items of requests $\{i_1,...,i_p\}$ hold the same placements in both plans.

Proof: Let j be the total number of items belonging to the first p identical requests $\{i_1,...,i_p\}$ of the request sequences rs_1 and rs_2. Then the corresponding item sequences is_1 and is_2 contain j identical items at the beginning (after getting ordered by Ord_u). When the heuristic H_v constructs the packing plan for both item sequences, obviously the first j identical items are getting the same placements because the heuristics do not look forward, i.e. to determine the placements for the first j items, only the properties (length and width) of these items are taken into account. The heuristics do not take into account the number, length or width of further items in is_1 and is_2 holding positions greater than j. Together with the fact that H_1–H_6 do not contain any stochastic components, this ensures identical placements for the first j items in both packing plans. □

With *Proposition* 4 it becomes clear, that to check a LRB in terms of packing for the 2L-PDP, it is sufficient to find a certain heuristic H_v and a certain ordering Ord_u such that the pair (H_v, Ord_u) can pack successfully the request sequences of all last pickup points of the considered LRB. Now the packing procedure *packing_check_rs* for the IPR routes can be enhanced to the procedure *packing_check_lrb* which checks a LRB in terms of packing for the 2L-PDP.

In Algorithm 3, the packing procedure *packing_check_lrb* is shown in detail. The procedure uses the cache *cache-lrb* to speed up the search. The procedure checks first if the input *lrb* is contained in *cache-lrb*. In this case, the result is taken from the cache and the procedure terminates. Otherwise, the procedure builds up the array *rs-arr* of all request sequences of last pickup points of *lrb*. Then the procedure tries to find a (u, v)-pair for which H_v can construct packing plans for all elements of *rs-arr* using the ordering Ord_u. For this purpose, the procedure uses three nested loops over the orderings, the heuristics and the item sequences. Each time a new ordering will be processed, the item sequence array *is-arr* will be built up. It contains the ordered item sequences for the request sequences of *rs-arr*. To speed up the packing procedure, this step will be done before entering the loop over the heuristics. In the most inner loop, the elements of *is-arr* are getting packing checked with heuristic H_v. The boolean variable *ok* indicates whether all item sequences could be packed successfully using the (H_v, Ord_u)-pair. In case of *ok=true* the procedure ends with *result*=true because packing plans fulfilling the reloading ban constraint for all last pickup points of *lrb* were found. Otherwise *ok=false* signals that at least one item sequence could not be packed with heuristic H_v using ordering Ord_u. In this case, the procedure continues with the next heuristic or the next ordering. If there are no further (H_v, Ord_u)-pairs remaining, the procedure ends with *result*=false. Before leaving the procedure, the overall packing result for *lrb* is saved in *cache-lrb*.

```
packing_check_lrb (in: lifo route block lrb, out: boolean result)
    if cache-lrb.contains(lrb) then
        return cache-lrb.get-result(lrb)
    rs-arr := build-request-sequences(lrb)          // build request sequences for all last pickup points in lrb
    for u := 1 to 5 do                              // loop over u (orderings)
        is-arr := { }                               // allocate empty array for item sequences
        for j := 1 to sizeof(rs-arr) do
            is-arr[j] := build-item-sequence(rs-arr[j], $Ord_u$)   // build corresponding item sequences for request seq.
        for v := 1 to 6 do                          // loop over v (heuristics)
            ok := true
            for j := 1 to sizeof(is-arr) do         // loop over j (item sequences)
                if heuristic $H_v$ can not pack item sequence is-arr[j] then
                    ok := false                     // set ok to false to neglect (u, v)-pair
                    break inner for loop            // break for loop over index j
            if ok = true then
                cache-lrb.set-result(lrb, true)     // $H_v$ + $Ord_u$ have checked all request-seq. with success
                return true                         // save positive result in cache-lrb and return
    cache-lrb.set-result(lrb, false)                // all (u, v)-pairs were tried without success
    return false                                    // save negative result in cache-lrb and return
```

Algorithm 3: Packing procedure *packing_check_lrb*.

Finally, it should be mentioned that the procedure *packing_check_lrb* does not save packing plans during the search process for already checked LRBs, because this would consume too much memory. In the *cache-lrb* (beside the LRB itself and its checking result) only the values u and v are saved, i.e. the numbers of the heuristic H_v and the ordering Ord_u which were able to construct feasible packing plans for all last pickup points of the considered LRB. When the stopping criterion in the LNS routing procedure is met and the search is getting aborted, the packing plans for all last pickup points in the best solution are reconstructed by the appropriate packing heuristic H_v in combination with the appropriate ordering Ord_u.

5 Computational experiments

The section is organized as follows. In the first part we test our solution approach against well-known 2L-CVRP instances to check whether we can reach the solution quality of the best so far existing algorithms for the 2L-CVRP. Since the 2L-CVRP is a special case of the 2L-PDP, where all pickups take place in the depot and after performing one delivery, no other pickup may follow in the route, our solutions approach can be used to solve 2L-CVRP instances, too. We use a modified version of the first hybrid algorithm (IPR) to meet the special requirements of the 2L-CVRP. In the 2L-CVRP there is to construct only one packing plan for each route with no LIFO constraint for pickup points (C1) and no reloading ban constraint (C3) to observe. Thus the two additional requirements to the routes (see section 4.2) can be dropped in this test, while LIFO constraint of the 2L-CVRP is ensured by the packing procedure.

In the second part, 60 new benchmark instances with up to 150 requests and 433 items are introduced. In the third part, which is the main part of this section, the two new solution approaches "Independent Partial Routes" and "Simultaneous Packing" are tested against the new instances. The test results will be compared with two lesser constrained problem variants, namely the "Unrestricted" and "One Dimensional" (1D). In the Unrestricted variant, we assume that at each node any reloading can be made without any cost or time consumption, hence both LIFO constraints (C1) and (C2) and the reloading ban constraint (C3) will be neglected in this variant. In the 1D variant, we drop furthermore the requirement to construct feasible packing plans, hence it is only required that the total area of all items loaded on the vehicle does not exceed the total area of the loading area, without considering if a feasible packing exists. Thus, the 1D variant is identical to the classical PDP with two scalar capacity conditions (weight and area). For both 1D variant and Unrestricted variant, we use a modified version of the IPR hybrid algorithm where the two additional requirements to the routes are dropped. In the 1D variant no packing checks will be done at all, while in the Unrestricted variant there will be constructed feasible packing plans for all nodes in a route. These packing plans do not need to meet the LIFO constraint for pickup points, thus the first criterion in all orderings (see section 4.1) is neglected in this variant.

Both hybrid algorithms are implemented in Java programming language using Eclipse IDE. All the experiments have been conducted on a PC with Intel Core i7-6700K (4.0 GHz, 32 GB RAM) running Windows 10 operating system. The identical chosen parameter setting for the routing procedure LNS of both hybrid algorithms is shown in Table 5. All parameter values were determined based on limited computational experiments using a trial and error strategy.

Table 5: Parameter setting for the LNS routing procedure.

Parameter	Description	Value
r_{min}	lower bound for no. of removed customers	$0.04 \cdot n$
r_{max}	upper bound for no. of removed customers	$0.4 \cdot n$
w	start temperature control parameter	0.005
c	rate of geometrical cooling	0.9999
$p(Rh_R)$, $p(Rh_S)$	probability of Random / Shaw removal	0.3, 0.4
$p(Rh_W)$, $p(Rh_T)$	probability of Worst / Tour removal	0.1, 0.2
$p(Ih_G)$, $p(Ih_{R2})$, $p(Ih_{R3})$	probability of Greedy / Regret-2 / Regret-3 insert	0.1, 0.6, 0.3
w_{r1}, w_{r2}	weights of relatedness formula for Shaw removal	9, 2

5.1 Computational results for 2L-CVRP

Gendreau et al. (2008) have introduced 36 CVRP problems containing up to 255 customers. For each of these CVRP problems, they created five 2L-CVRP instances by considering different classes of item characteristics (Class 1–5). Thereby, the instances of Class 1 are pure CVRP problems, whereas the 144 instances of Classes 2–5 are "real" 2L-CVRP problems. For these instances, each item belongs to one of three possible shape categories with equal probability, while the number of items demanded by a customer is determined as random value from a given interval. The details of the items characteristics are shown in Table 6. The average number of items per customer rises from 1.5 for instances of Class 2 up to 3 items per customer for instances of Class 5.

Table 6: The item characteristics for 2L-CVRP instances of Class 2–5.

Class	Item-Number m_i	Vertical		Homogeneous		Horizontal	
		Length	Width	Length	Width	Length	Width
2	[1, 2]	[0.4L, 0.9L]	[0.1W, 0.2W]	[0.2L, 0.5L]	[0.2W, 0.5W]	[0.1L, 0.2L]	[0.4W, 0.9W]
3	[1, 3]	[0.3L, 0.8L]	[0.1W, 0.2W]	[0.2L, 0.4L]	[0.2W, 0.4W]	[0.1L, 0.2L]	[0.3W, 0.8W]
4	[1, 4]	[0.2L, 0.7L]	[0.1W, 0.2W]	[0.1L, 0.4L]	[0.1W, 0.4W]	[0.1L, 0.2L]	[0.2W, 0.7W]
5	[1, 5]	[0.1L, 0.6L]	[0.1W, 0.2W]	[0.1L, 0.3L]	[0.1W, 0.3W]	[0.1L, 0.2L]	[0.1W, 0.6W]

Our hybrid algorithm, in the following denoted with "LNS+6CH", was tested five times against each of the 144 instances considering the problem variant with LIFO constraint and fixed orientation ("NoRotate") allowing up to one hour CPU time. The results are first averaged over the five runs and then averaged over the Classes 2–5 for each of the 36 problems. In Table 7, the results for the LNS+6CH hybrid algorithm are compared to those obtained by

- ACO with 3 hours CPU time (Fuellerer et al., 2009), Pentium IV 3.2 GHz,
- EGTS + LBFH (Leung et al., 2011), Intel Core 2 Duo 2.0 GHz,
- PRMP (Zachariadis et al., 2013), Intel Core 2 Duo E6600 2.4 GHz,
- VNS (Wei et al., 2015), Intel Xeon E5430 with a 2.66 GHz.

Table 7: Results for 2L-CVRP (averaged over Classes 2–5).

Prob-lem	ACO			EGTS + LFBH			PRMP			VNS			LNS+6CH		
	ttd-avg	rt-avg	gap %	ttd-avg	rt-avg	gap %	ttd-avg	rt-avg	gap %	ttd-avg	rt-avg	gap %	ttd-avg	rt-avg	gap %
1	295.10	8.9	2.79	303.40	3.2	5.68	**287.09**	1.4	0.00	287.52	4.5	0.15	298.01	5.6	3.80
2	345.28	0.7	0.31	345.23	1.3	0.30	**344.21**	1.0	0.00	**344.21**	0.6	0.00	345.23	5.0	0.30
3	383.12	3.7	0.45	387.89	5.0	1.70	**381.40**	1.3	0.00	381.43	1.5	0.01	384.58	5.0	0.83
4	441.11	3.6	0.26	443.25	2.4	0.75	**439.97**	1.6	0.00	440.27	1.0	0.07	442.14	5.0	0.49
5	383.21	15.7	0.21	387.60	4.6	1.36	**382.39**	2.6	0.00	**382.39**	4.2	0.00	386.80	6.0	1.15
6	500.76	5.2	0.26	502.25	3.5	0.55	**499.48**	5.6	0.00	**499.48**	1.7	0.00	502.88	5.0	0.68
7	705.64	14.3	0.49	715.54	8.3	1.90	702.27	5.3	0.01	**702.18**	12.3	0.00	711.31	5.4	1.30
8	713.33	19.9	2.26	716.36	8.3	2.69	699.55	7.0	0.28	**697.58**	21.2	0.00	713.12	7.4	2.23
9	616.69	5.8	0.28	621.23	4.3	1.02	615.93	6.2	0.16	**614.95**	3.7	0.00	615.94	5.0	0.16
10	701.65	61.6	2.01	731.69	23.3	6.38	688.63	55.0	0.12	**687.80**	115.8	0.00	711.32	20.5	3.42
11	736.53	71.5	1.57	762.83	38.0	5.20	725.83	75.3	0.10	**725.11**	54.5	0.00	748.82	23.8	3.27
12	617.07	8.6	0.41	622.35	7.9	1.27	615.23	7.1	0.12	**614.52**	7.5	0.00	620.41	5.1	0.96
13	2598.46	76.3	1.75	2647.88	30.8	3.69	2554.93	119.6	0.05	**2553.76**	53.5	0.00	2617.70	25.4	2.50
14	1050.48	202.3	1.93	1075.04	49.0	4.31	**1030.61**	637.1	0.00	1033.12	416.5	0.24	1054.56	67.4	2.32
15	1223.03	172.5	3.09	1223.19	65.8	3.10	1193.88	68.2	0.63	**1186.38**	298.5	0.00	1210.15	57.5	2.00
16	701.72	9.4	0.10	703.74	13.0	0.39	**701.01**	14.2	0.00	**701.01**	3.4	0.00	704.75	5.0	0.53
17	865.56	7.5	0.17	869.93	16.1	0.68	865.33	40.9	0.15	**864.06**	4.4	0.00	864.57	5.0	0.06
18	1073.34	362.1	1.52	1096.57	67.8	3.72	1061.29	95.1	0.38	**1057.27**	396.5	0.00	1078.03	120.7	1.96
19	779.29	149.8	2.05	798.20	61.7	4.53	767.13	188.3	0.46	**763.62**	297.5	0.00	779.68	49.9	2.10
20	544.79	1322.6	2.34	559.17	232.5	5.05	535.89	1660.9	0.67	**532.31**	921.8	0.00	544.18	440.9	2.23
21	1061.44	1284.6	2.63	1084.98	179.0	4.91	1043.12	420.2	0.86	**1034.25**	1003.0	0.00	1055.95	428.2	2.10
22	1086.84	904.9	2.45	1113.64	152.3	4.97	1068.35	524.3	0.71	**1060.87**	1107.8	0.00	1085.37	301.6	2.31
23	1100.68	1490.3	2.96	1130.13	215.3	5.72	1080.59	519.5	1.09	**1068.99**	953.0	0.00	1094.16	496.8	2.35
24	1158.06	389.6	2.03	1177.28	132.7	3.72	1143.88	1064.3	0.78	**1135.05**	841.3	0.00	1157.30	129.9	1.96
25	1428.74	3007.9	3.07	1470.11	373.2	6.06	1403.33	2319.5	1.24	**1386.16**	1306.0	0.00	1423.94	1002.6	2.73
26	1427.91	4379.0	4.98	1431.32	499.0	5.23	1374.49	1491.2	1.05	**1360.19**	1240.3	0.00	1387.02	1459.4	1.97
27	1400.46	1898.3	2.79	1445.64	371.4	6.10	1378.13	4163.8	1.15	**1362.50**	1242.3	0.00	1396.34	632.8	2.48
28	2734.60	10800.8	2.70	2808.10	979.8	5.46	2677.71	8640.1	0.57	**2662.59**	2423.3	0.00	2683.39	3600.0	0.78
29	2361.32	10800.9	4.80	2396.78	1150.4	6.37	2273.25	5484.3	0.89	**2253.26**	2672.8	0.00	2309.20	3600.0	2.48
30	1906.16	10800.7	4.10	1983.48	1699.0	8.32	1858.69	4676.9	1.51	**1831.09**	2502.0	0.00	1876.39	3600.0	2.47
31	2431.13	10800.8	4.18	2497.25	4368.2	7.02	2370.77	5845.4	1.59	**2333.55**	2760.8	0.00	2390.12	3600.0	2.42
32	2400.06	10800.6	4.96	2438.65	2445.8	6.65	2332.28	9433.2	2.00	**2286.62**	2664.0	0.00	2337.85	3600.0	2.24
33	2467.61	10800.6	4.72	2543.24	2053.7	7.93	2404.52	5662.5	2.04	**2356.37**	2614.5	0.00	2413.23	3600.0	2.41
34	1272.29	10800.7	5.84	1276.27	3443.5	6.17	1231.90	13141.8	2.48	**1202.10**	2825.8	0.00	1241.73	3600.0	3.30
35	1612.42	10800.7	10.00	1606.38	4560.8	9.59	1500.97	8989.6	2.40	**1465.77**	3053.0	0.00	1593.32	3600.0	8.70
36	1846.66	10800.8	5.39	1850.50	3667.1	5.61	1774.94	10059.6	1.30	**1752.16**	3282.5	0.00	1891.36	3600.0	7.94
Avg			**2.55**			**4.28**			**0.69**			**0.01**			**2.25**

The allowed CPU time for LNS+6CH is set to one third of the CPU time consumed by the ACO algorithm of Fuellerer et al. (2009) but not less than 5 seconds. For each algorithm, the average total travel distance and average computation time (averaged over Classes 2–5) are provided. For each problem, the minimum average total travel distance value is marked in bold. The gap in percent is calculated as (*avg-ttd / min-avg-ttd* – 1)*100%.

The results show that LNS+6CH performs better than the “older” algorithms ACO (0.30%) and EGTS + LFBH (2.03%) but cannot reach the solution quality of the “newer” algorithms PRMP (1.56%) and VNS (2.24%). Nevertheless, these gaps are small and the computation times used for LNS+6CH are comparable to those of the other algorithms, so LNS+6CH can be considered as “state of the art” procedure for solving the 2L-CVRP.

5.2 Benchmark instances for 2L-PDP

The 60 new 2L-PDP instances were generated based on the 2L-CVRP instances by Gendreau et al. (2008). First 20 2L-CVRP instances with 25 to 150 customers were selected as shown in Table 8.

Table 8: 2L-CVRP instances used to create new 2L-PDP benchmark instances.

Average item count per request	**Customer count**				
	25	**50**	**75**	**100**	**150**
1.5	09-2	19-2	21-2	25-2	30-2
2.0	09-3	19-3	21-3	25-3	30-3
2.5	09-4	19-4	21-4	25-4	30-4
3.0	09-5	19-5	21-5	25-5	30-5

For each of the selected 2L-CVRP instances, three 2L-PDP instances were created with different characteristics regarding the distribution of pickup and delivery points of the requests. In the first variant "Random", the sites are uniformly distributed in a rectangular section of the plane, while they are clustered in the other variants. In the second variant "Mixed clusters", individual clusters may contain pickup as well as delivery points, while only sites of one sort can occur in an individual cluster of the third variant "Pure clusters". The three types of 2L-PDP instances are denoted by the suffixes “-Rnd”, “-Mix” and “-Pur”, e.g. the instances constructed based on 09-2 are denoted with 09-2-Rnd, 09-2-Mix and 09-2-Pur.

As in the original 2L-CVRP instances, each four 2L-PDP instances for the same problem number and the same distribution variant (e.g. 09-2-Rnd to 09-5-Rnd) share the same node set. The item sets, the requests weights and the dimension of the loading areas (L = 40, W = 20) were taken over from the original instances without any change (see Table 9). The vehicles weight capacity was slightly adapted to ensure that the weight constraint (C4) does not become redundant but a good utilization of the loading area is still possible and the packing task is not too easy. The maximum route length was defined so that best solutions found by the hybrid algorithms contain a reasonable number of routes (from 3 to

15 routes depending on the size of the instance). The characteristics of the 2L-PDP instances are summarized in Table 9. The instances are offered at the website
http://www.mansci.ovgu.de/Forschung/Materialien.html.

Table 9: Overview of the new 2L-PDP benchmark instances.

Parameter	Value	Remark
Total number of instances	60	
Request number per instance	25 / 50 / 75 / 100 / 150	12 instances each
Node number per instance	50 / 100 / 150 / 200 / 300	12 instances each
Average number of items per request	1.5 / 2.0 / 2.5 / 3.0	15 instances each
Distribution variants of pickup / delivery points	Random / Pure Cluster / Mixed Cluster	20 instances each

5.3 Computational results for the 2L-PDP

The detailed results for the 2L-PDP instances regarding total travel distance (*ttd*) are presented in Table 10 and 11. The structure for both tables is identical, Table 10 covers the "Rotate" variant where 90° rotations of the items are allowed, while Table 11 corresponds to the "NoRotate" variant with the additional constraint (C7).

Table 10: Results (travel distances) for different variants of 2L-PDP ("Rotate" variant).

Instance				1D	Unrestricted			Simultaneous Packing			Independent Partial Routes		
type	req. n	items m	CPU sec	avg-ttd	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort
09-2	25	40	30	786.63	823.83	4.70	156.62	962.75	22.97	-	1007.82	28.41	-
09-3	25	61	30	812.82	849.68	4.58	158.31	971.18	19.95	-	999.83	23.25	-
09-4	25	63	30	801.42	845.59	5.37	145.94	995.88	24.76	-	1017.64	27.33	-
09-5	25	91	30	772.90	785.15	1.54	185.65	900.69	16.77	-	975.16	25.96	-
19-2	50	82	60	1215.46	1283.19	5.53	175.56	1527.66	26.03	-	1595.75	31.31	-
19-3	50	103	60	1257.50	1299.03	3.30	183.71	1548.52	23.43	-	1603.02	27.67	-
19-4	50	134	60	1277.66	1339.93	4.91	178.30	1580.34	23.70	-	1655.35	29.44	-
19-5	50	157	60	1113.60	1136.04	1.99	268.14	1410.24	26.86	-	1495.82	33.92	-
21-2	75	114	120	1659.17	1745.65	5.45	201.10	2086.47	25.66	-	2113.70	27.40	-
21-3	75	164	120	1845.22	1952.08	5.72	182.00	2236.64	20.96	-	2259.65	22.28	-
21-4	75	168	120	1683.96	1735.30	2.98	191.06	2098.60	24.43	-	2126.25	26.06	-
21-5	75	202	120	1560.33	1596.65	2.39	258.28	1970.56	26.13	-	2024.57	29.64	-
25-2	100	157	300	2254.38	2398.23	6.39	176.77	2878.46	27.57	-	2944.73	30.60	-
25-3	100	212	300	2258.33	2348.21	3.97	204.74	2844.93	25.84	-	2909.35	28.81	-
25-4	100	254	300	2274.79	2350.94	3.36	190.97	2842.27	24.92	-	2913.71	28.19	-
25-5	100	311	300	2009.49	2043.85	1.71	271.30	2654.88	31.81	-	2736.86	36.05	-
30-2	150	225	900	3018.56	3169.07	5.00	191.66	3844.96	27.32	-	3900.22	29.20	-
30-3	150	298	900	3182.12	3313.98	4.15	190.92	3958.25	24.37	-	4027.43	26.60	-
30-4	150	366	900	3144.45	3251.56	3.41	194.16	3913.52	24.43	-	3953.07	25.74	-
30-5	150	433	900	2772.60	2821.62	1.72	274.09	3540.44	27.61	-	3613.64	30.34	-
Average						**3.91**	**198.96**		**24.78**			**28.41**	

Table 11: Results (travel distances) for different variants of 2L-PDP ("NoRotate" variant).

Instance				1D	Unrestricted			Simultaneous Packing			Independent Partial Routes		
type	req. n	items m	CPU sec	avg-ttd	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort
09-2	25	40	30	786.63	888.97	12.77	127.93	1015.79	29.61	-	1049.79	33.72	-
09-3	25	61	30	812.82	894.75	9.79	153.85	1021.14	25.96	-	1060.53	30.56	-
09-4	25	63	30	801.42	892.24	10.93	106.92	1013.19	26.90	-	1027.84	28.63	-
09-5	25	91	30	772.90	785.12	1.59	180.01	901.34	16.85	-	979.45	26.51	-
19-2	50	82	60	1215.46	1340.42	10.41	179.32	1594.17	31.72	-	1651.59	36.14	-
19-3	50	103	60	1257.50	1347.99	7.27	180.90	1609.69	28.31	-	1653.40	31.92	-
19-4	50	134	60	1277.66	1371.68	7.40	162.01	1628.44	27.71	-	1684.76	31.88	-
19-5	50	157	60	1113.60	1152.01	3.43	231.37	1437.16	29.42	-	1534.23	37.52	-
21-2	75	114	120	1659.17	1810.71	9.27	203.86	2172.99	30.97	-	2219.16	33.90	-
21-3	75	164	120	1845.22	1993.00	7.97	177.08	2311.73	25.08	-	2338.55	26.57	-
21-4	75	168	120	1683.96	1771.38	5.12	170.97	2140.80	26.98	-	2172.33	28.86	-
21-5	75	202	120	1560.33	1632.27	4.70	229.42	2004.40	28.31	-	2043.14	30.89	-
25-2	100	157	300	2254.38	2526.46	12.11	170.78	3005.40	33.20	-	3088.52	36.98	-
25-3	100	212	300	2258.33	2443.66	8.19	192.45	2963.90	31.14	-	3025.99	33.99	-
25-4	100	254	300	2274.79	2406.40	5.78	175.17	2929.75	28.80	-	2988.75	31.53	-
25-5	100	311	300	2009.49	2075.81	3.31	233.85	2683.32	33.21	-	2770.52	37.70	-
30-2	150	225	900	3018.56	3315.24	9.88	171.31	3990.33	32.15	-	4028.47	33.45	-
30-3	150	298	900	3182.12	3434.56	7.94	177.21	4094.93	28.69	-	4148.72	30.42	-
30-4	150	366	900	3144.45	3344.72	6.38	185.50	4048.24	28.72	-	4084.07	29.93	-
30-5	150	433	900	2772.60	2865.08	3.29	243.84	3602.58	29.87	-	3668.91	32.38	-
Average						**7.38**	**182.69**		**28.68**			**32.17**	

In the leftmost column of both tables, the instance types are listed. The next three columns show the number of requests, the number of items and the allowed CPU time, which varies from 30 to 900 seconds depending on the size of the instance (apart from that, the allowed computation time for the 1D variant is set to only 20% of the normal value). The fifth column shows the total travel distances for the 1D variant for which only the weight constraint (C4), and the routing constraints (C5) and (C6) are considered and no packing check will be done (only the total item area will still be checked). In the following nine columns, the total travel distances, the gaps and the reloading quantities are indicated for the Unrestricted variant and also for the two hybrid algorithms (Simultaneous Packing / Independent Partial Routes) for the original problem variant. In the Unrestricted, variant the constraints (C1) – (C3) are omitted while the others constraints, especially the need to find valid packing plans, are still in force. In this variant reloading effort at each pickup or delivery point can occur, i.e. temporary or permanent changes of placements of items which do not belong to the loaded / unloaded request may happen. In the Simultaneous Packing and Independent Partial Routes solution approach all constraints are in force as described in Chapter 3, where all reloading effort is ruled out by the constraints (C1) – (C3). In the Simultaneous Packing approach, the reloading ban constraint is enforced by a new type of packing procedure, while in the Independent Partial Routes approach the reloading ban constraint (C3) is enforced by an additional routing condition instead of the simultaneous packing checks which reduces the numerical effort but restricts the search space more. All presented total travel distances are mean values over five runs. To keep the tables compact the results are averaged, furthermore, over all instances of the same type, e.g. "09-2" stands for the three 2L-PDP instances which are derived from the original 2L-CVRP instance 09-2. The corresponding gaps are calculated as ($ttd - ttd_{1D}$) / ttd_{1D} * 100 (%). The reloading effort is given as percentage of the total item area (= sum of the area of all items in the instance). If an item is reloaded, say, at three nodes in the route, then the area of the item is counted three times. Thus it may occur that the reloading effort exceeds 100%. In the last lines of Tables 10 and 11, the gap values of the 2L-PDP variants are averaged over the 60 instances. Detailed results for each single instance are presented in Tables 13 and 14 of appendix A.

Summarizing the results for the "Rotate" problem variant, we can state that the travel distances increase significantly increases if the 2L-PDP instances are solved instead of the corresponding 1D-PDP instances. For the Unrestricted variant, the total travel distances grow on average by 3.91% compared to the 1D case. For the original problem variant, the mean gap is even higher and amounts to 24.78% (Simultaneous Packing approach) and 28.41% (Independent Partial Routes approach), respectively. For the Unrestricted variant arises a reloading effort of 198.96% on average, which means that each item was reloaded (on average) nearly two times during its route, while for the two new hybrid algorithms for the original problem variant no reloading effort occurs by definition. So we come to the conclusion that avoiding any reloading effort leads to increase of the travel costs of approximately 20% or the other way round, we can save approximately 20% of the travel costs if we are willing to pay this in form of the additional reloading effort. The comparison between the two new hybrid algorithms shows that the more complex Simultaneous Packing approach performs 2.83% (124.78% to 128.41%) better than the simpler Independent Partial Routes approach if no reloading is allowed. This result coincides with the expectation formulated in section 4 (see Table 1).

For the "NoRotate" problem variant, the results regarding total travel distance show gaps which are approximately 4% points larger than in the "Rotate" variant. This result is plausible because the packing task without the possibility to rotate items is more difficult to solve. This leads to an additional restriction of the solution space and an increase of the best objective function value. In case of omitting the constraints (C1) – (C3), the "NoRotate" problem variant shows a smaller reloading effort (182.69% to 198.96%) because the longer routes lead to generally "less occupied" loading areas and to less situations where reloading effort can occur. Furthermore, the results of the "NoRotate" variant confirm the conclusions we made in the previous paragraph.

In Table 12, the average computing times to find the best solution and the average total number of iterations executed are shown for the two new hybrid algorithms. Again the results are averaged over all instances of the same type, while the detailed results are presented in Table 15 of Appendix A. The times are given as absolute values and as percentages of the allowed computing time per instance. In the last column the ratio of executed iterations of both hybrid algorithms is shown ($iterations_{SP}$ / $iterations_{IPR}$). All values are averaged over five runs. The results show that the simpler Independent Partial

Routes approach only needs 44.78% of the computing time on average to find the best solution while the Simultaneous Packing approach needs 48.63% to find the best solution. The comparison of total executed iterations shows that the Simultaneous Packing approach can execute only approximately 40% of the iterations of the other approach in the same computing time. This shows that the Simultaneous Packing approach is more expensive in terms of CPU usage than the Independent Partial Routes approach because of the more complex packing algorithm. On the other hand, there may be still potential for further improvements with the Simultaneous Packing approach if more CPU time would be allowed (especially for the instances with 150 requests). Again, this result coincide with the expectation formulated in section 4 (see Table 1).

Table 12: Total iteration numbers and computing times to find the best solution ("Rotate" variant).

Instance				**Independent Partial Routes**			**Simultaneous Packing**			**Ratio iterations**
type	**req. n**	**items m**	**CPU sec**	**Runtime to best**	**Runtime to best in %**	**Total iterations**	**Runtime to best**	**Runtime to best in %**	**Total iterations**	
09-2	25	40	30	3.93	13.10	743407.67	3.38	11.27	316799.00	0.43
09-3	25	61	30	5.57	18.58	760754.67	4.41	14.69	325637.33	0.43
09-4	25	63	30	5.04	16.79	737302.00	4.85	16.16	322605.67	0.44
09-5	25	91	30	4.69	15.63	744638.00	1.75	5.83	281710.67	0.38
19-2	50	82	60	20.80	34.67	318347.67	18.74	31.23	121372.67	0.38
19-3	50	103	60	16.93	28.21	319906.33	24.49	40.82	125209.33	0.39
19-4	50	134	60	19.87	33.12	312806.33	23.58	39.29	124830.00	0.40
19-5	50	157	60	12.88	21.46	267537.00	17.10	28.49	95587.33	0.36
21-2	75	114	120	56.77	47.31	228759.33	69.26	57.72	85333.33	0.37
21-3	75	164	120	70.48	58.73	250279.67	70.77	58.98	101774.67	0.40
21-4	75	168	120	66.60	55.50	204657.00	73.52	61.26	71585.67	0.35
21-5	75	202	120	59.60	49.67	178960.33	71.32	59.43	63674.33	0.35
25-2	100	157	300	177.60	59.20	357548.33	195.71	65.24	141081.33	0.39
25-3	100	212	300	179.42	59.81	347215.00	198.79	66.26	141376.00	0.41
25-4	100	254	300	184.13	61.38	330252.00	160.52	53.51	137278.00	0.42
25-5	100	311	300	169.64	56.55	265729.67	181.00	60.33	103366.33	0.39
30-2	150	225	900	626.13	69.57	389412.33	661.43	73.49	152337.67	0.39
30-3	150	298	900	630.90	70.10	378192.67	725.95	80.66	163377.33	0.43
30-4	150	366	900	634.83	70.54	354757.67	717.30	79.70	149151.33	0.42
30-5	150	433	900	500.48	55.61	286010.67	614.65	68.29	111248.33	0.39
Average					**44.78**			**48.63**		**0.40**

6 Conclusions and future work

In this paper, the vehicle routing problem with pickup and delivery (PDP) has been extended to an integrated vehicle routing and loading problem with 2D rectangular items to be transported in homogeneous vehicles on a rectangular 2D loading area (2L-PDP). In the problem formulation, we focused on the question under which conditions any reloading effort, i.e. any movement of items after loading and before unloading, can be avoided. It turned out that the LIFO constraints for pickup and delivery points are not sufficient. Instead, the new reloading ban constraint was required to rule out any reloading effort.

Two solution approaches implemented as hybrid algorithms consisting of a routing and a packing procedure were proposed to tackle the 2L-PDP. In the first solution approach (Independent Partial Routes), a large neighborhood search procedure for routing is combined with a packing procedure using six well-known constructive packing heuristics. To ensure the LIFO constraint at delivery points and the reloading ban constraint the search space must be restricted to routes which are fulfilling two additional requirements (1) and (2) (see Section 4.2). In the second more complex solution approach (Simultaneous Packing), basically the same routing procedure is combined with a new type of packing procedure which is able to construct a series of interrelated packing plans fulfilling the reloading ban constraint (see Section 4.4). Therefore, in the second approach the additional requirement (1) to the routes can be dropped.

The hybrid algorithms were tested with the well-known 2L-CVRP instances by Gendreau et al. (2008) and reached a good solution quality compared to the best 2L-CVRP solution methods available. For testing the hybrid 2L-PDP algorithms, 60 2L-PDP instances with up to 150 requests and up to 433 items were introduced. The results for the 2L-PDP variants are plausible in that the second approach performs nearly 3% better than the first solution approach on average. Neglecting LIFO and reloading ban constraints (Unrestricted variant) would lead to a reduction of around 20% of the total travel distance. Put differently, ruling out any reloading has to be paid by a 20% increase of travel distance.

In future research, a packing procedure based on the second solution approach should be developed, which is able to observe the LIFO constraint for delivery points, too. This would allow to drop also the additional requirement (2) so that a further improvement of the solution quality could be expected.

References

Bartók, T; Imre, C (2011): Pickup and Delivery Vehicle Routing with Multidimensional Loading Constraints. *Acta Cybernetica*, 20, 17–33.

Bent, R; van Hentenryck, P (2006): A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows. *Computers & Operations Research*, 33:875–893.

Berbeglia, G; Cordeau, JF; Gribkovskaia, I; Laporte, G (2007): Static pickup and delivery problems: A classification scheme and survey. *Top*, 15:1–31.

Bortfeldt, A (2012): A Hybrid Algorithm for the Capacitated Vehicle Routing Problem with Three-Dimensional Loading Constraints. *Computers & Operations Research*, 39:2248–2257.

Bortfeldt, A; Homberger, J (2013): Packing First, Routing Second - a Heuristic for the Vehicle Routing and Loading Problem. *Computers & Operations Research*, 40:873–885.

Bortfeldt, A; Hahn, T; Männel, D; Mönch, L (2015): Hybrid algorithms for the vehicle routing problem with clustered backhauls and 3D loading constraints. *European Journal of Operational Research*, 243:82–96.

Chazelle, B (1983): The bottom-left bin packing heuristic: An efficient implementation. *IEEE Transactions on Computers*, C-32:697–707.

Crainic, T; Perboli, G; Tadei, R (2008): Extreme Point-based Heuristics for Three-dimensional Bin Packing. *INFORMS Journal on Computing*, 20:368–384.

Dominguez, O; Guimarans, D; Juan, AA; De La Nuez, I (2016): A Biased-Randomised Large Neighbourhood Search for the two-dimensional Vehicle Routing Problem with Backhauls. *European Journal of Operational Research*, 255:442-462.

Duhamel, C; Lacomme, P; Quilliot, A; Toussaint, H (2011): A multi-start evolutionary local search for the two-dimensional loading capacitated vehicle routing problem. *Computers & Operations Research*, 38:617–640.

Fuellerer, G; Doerner, KF; Hartl, RF; Iori, M (2009): Ant colony optimization for the two-dimensional loading vehicle routing problem. *Computers & Operations Research*, 36:655–673.

Fuellerer, G; Doerner, KF; Hartl, R; Iori, M (2010): Metaheuristics for Vehicle Routing Problems with Three-dimensional Loading Constraints. *European Journal of Operational Research*, 201:751–759.

Gendreau, M; Iori, M; Laporte, G; Martello, S (2006): A Tabu Search Algorithm for a Routing and Container Loading Problem. *Transportation Science*, 40:342–350.

Gendreau, M; Iori, M; Laporte, G; Martello, S (2008): A tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. *Networks,* 51:4–18.

Gendreau, M; Potvin, JY (2010): Handbook of Metaheuristics, second edition. Springer, New York.

Iori, M; Salazar Gonzalez, JJ; Vigo, D (2007): An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transportation Science*, 41:253–264.

Iori, M; Martello, S (2010): Routing Problems with Loading Constraints. *Top*, 18:4–27.

Iori, M; Martello, S (2013): An Annotated Bibliography of Combined Routing and Loading Problems. *Yugoslav Journal of Operations Research*, 23(3):311–326.

Khebbache-Hadji, S; Prins, C; Yalaoui, A; Reghioui, M (2013): Heuristics and memetic algorithm for the two-dimensional loading capacitated vehicle routing problem with time windows. *Central European Journal of Operations Research,* 21:307–336.

Leung, SCH; Zhou, X; Zhang, D; Zheng, J (2011): Extended guided tabu search and a new packing algorithm for the two-dimensional loading vehicle routing problem. *Computers & Operations Research*, 38:205–215.

Leung, SCH; Zhang, Z; Zhang, D; Hua, X; Lim, MK (2013): A meta-heuristic algorithm for heterogeneous fleet vehicle routing problems with two-dimensional loading constraints. *European Journal of Operational Research*, 225:199–210.

Li, H; Lim, A (2001): A metaheuristic for the pickup and delivery problem with time windows. 13th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'01). IEEE Computer Society, Los Alamitos, CA, 333–340.

Lodi, A; Martello, S; Vigo, D (1999): Heuristic and Metaheuristic Approaches for a Class of Two-Dimensional Bin Packing Problems. *INFORMS Journal on Computing*, 11:345–357.

Lu, Q; Dessouky, MM (2006): A new insertion-based construction heuristic for solving the pickup and delivery problem with time windows. *European Journal of Operational Research*, 175:672–687.

Malapert, A; Guéret, C; Jussien, N; Langevin, A; Rousseau, LM (2008): Two-dimensional Pickup and Delivery Routing Problem with Loading Constraints. *Proceedings of the First CPAIOR Workshop on Bin Packing and Placement Constraints (BPPC'08)*, Paris, France.

Männel, D; Bortfeldt, A (2016): A Hybrid Algorithm for the Vehicle Routing Problem with Pickup and Delivery and Three-dimensional Loading Constraints. *European Journal of Operational Research*, 254:840–858.

Männel, D; Bortfeldt, A (2017): Solving the Pickup and Delivery Problem with Three-dimensional Loading Constraints and Reloading Ban. *European Journal of Operational Research*, in press.

Moura, A; Oliveira, JF (2009): An Integrated Approach to Vehicle Routing and Container Loading Problems. *Operations Research Spectrum* 31:775-800.

Nagata, Y; Kobayashi, S (2010): A Memetic Algorithm for the Pickup and Delivery Problem with Time Windows Using Selective Route Exchange Crossover. *Parallel Problem Solving from Nature, PPSN XI*, R. Schaefer et al. (eds.), 536-545, Springer: Berlin.

Nanry, WP; Barnes, W (2000): Solving the pickup and delivery problem with time windows using reactive tabu search. *Transportation Research Part B: Methodological*, 34:107–121.

Pankratz, G (2005): A grouping algorithm for the pickup and delivery problem with time windows. *OR Spectrum*, 27:21-41.

Parragh, SN; Doerner, KF; Hartl, RF (2008): A Survey on Pickup and Delivery Problems. Part II: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, 58:81–117.

Pollaris, H; Braekers, K; Caris, A; Janssens, G; Limbourg, S (2015): Vehicle routing problems with loading constraints: state-of-the-art and future directions. *OR Spectrum*, 37:297–330.

Ropke, S; Pisinger, D (2006): An Adaptive Large Neighborhood Search for the Pickup and Delivery Problem with Time Windows. *Transportation Science*, 40:455–472.

Ruan, Q; Zhang, Z; Miao, L; Shen, H (2013): A Hybrid Approach for the Vehicle Routing Problem with Three-dimensional Loading Constraints. *Computers & Operations Research,* 40:1579–1589.

Tao, Y; Wang, F (2015): An effective tabu search approach with improved loading algorithms for the 3L-CVRP. *Computers & Operations Research*, 55:127–140.

Tarantilis, CD; Zachariadis, EE; Kiranoudis, CT (2009): A Hybrid Metaheuristic Algorithm for the Integrated Vehicle Routing and Three-dimensional Container-loading Problem. *IEEE Transactions on Intelligent Transportation Systems,* 10:255–271.

Toth, P; Vigo, D (2014): Vehicle Routing: Problems, Methods, and Applications, second edition. MOS-SIAM series on optimization, Philadelphia.

Wang, L; Guo, S; Chen, S; Zhu, W; Lim, A (2010): Two Natural Heuristics for 3D Packing with Practical Loading Constraints. *Computer Science,* Vol. 6230, 256–267.

Wei, L; Zhang, Z; Lim, A (2014): An adaptive variable neighborhood search for a heterogeneous fleet vehicle routing problem with three-dimensional loading constraints. *IEEE Computational Intelligence Magazine* 9, 18–30.

Wei, L; Zhang, Z; Zhang, D; Lim, A (2015): A variable neighborhood search for the capacitated vehicle routing problem with two-dimensional loading constraints. *European Journal of Operational Research*, 243:798–814.

Wisniewski, M; Ritt, M; Buriol, LS (2011): A Tabu Algorithm for the Capacitated Vehicle Routing Problem with Three-dimensional Loading Constraints. *Anais do XLIII Simpósio Brasileiro de Pesquisa Operacional*. Ubatuba, Brazil, 1502–1511.

Zachariadis, EE; Tarantilis, CD, Kiranoudis, CT (2009): A Guided Tabu Search for the Vehicle Routing Problem with two-dimensional loading constraints. *European Journal of Operational Research*, 195:729–743.

Zachariadis, EE; Tarantilis, CD; Kiranoudis, CT (2012): The pallet-packing vehicle routing problem. *Transportation Science*, 46:341–358.

Zachariadis, EE; Tarantilis, CD, Kiranoudis, CT (2013): Integrated distribution and loading planning via a compact metaheuristic algorithm. *European Journal of Operational Research*, 228:56–71.

Zachariadis, EE; Tarantilis, CD, Kiranoudis, CT (2016): The Vehicle Routing Problem with Simultaneous Pick-ups and Deliveries and Two-Dimensional Loading Constraints. *European Journal of Operational Research*, 251:369-386.

Zhang, Z; Wei, L; Lim, A (2015): An evolutionary local search for the capacitated vehicle routing problem minimizing fuel consumption under three-dimensional loading constraints. *Transportation Research Part B*, 82:20–35.

Zhu, W; Qin, H; Lim, A; Wang, L (2012): A two-stage Tabu Search Algorithm with Enhanced Packing Heuristics for the 3L-CVRP and M3L-CVRP. *Computers & Operations Research,* 39:2178–2195.

Appendix A

Table 13: Results (travel distances) for different variants of 2L-PDP (“Rotate” variant, complete results).

Instance				1D	Unrestricted			Simultaneous Packing			Independent Partial Routes		
name	req. n	items m	CPU sec	avg-ttd	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort
09-2-Rnd	25	40	30	704.24	733.44	4.15	220.87	944.26	34.08	–	944.43	34.11	–
09-2-Mix	25	40	30	828.46	885.71	6.91	154.00	1025.23	23.75	–	1098.42	32.59	–
09-2-Pur	25	40	30	827.21	852.33	3.04	95.01	918.75	11.07	–	980.59	18.54	–
09-3-Rnd	25	61	30	727.29	755.03	3.81	167.18	944.26	29.83	–	949.75	30.59	–
09-3-Mix	25	61	30	881.69	892.55	1.23	167.03	1034.77	17.36	–	1115.25	26.49	–
09-3-Pur	25	61	30	829.47	901.47	8.68	140.71	934.50	12.66	–	934.50	12.66	–
09-4-Rnd	25	63	30	709.10	730.40	3.00	174.54	945.28	33.31	–	945.28	33.31	–
09-4-Mix	25	63	30	847.03	905.10	6.86	145.93	1045.78	23.46	–	1111.18	31.19	–
09-4-Pur	25	63	30	848.14	901.27	6.26	117.35	996.58	17.50	–	996.46	17.49	–
09-5-Rnd	25	91	30	706.07	709.09	0.43	205.65	869.40	23.13	–	869.40	23.13	–
09-5-Mix	25	91	30	821.05	833.10	1.47	155.85	955.09	16.33	–	1098.32	33.77	–
09-5-Pur	25	91	30	791.58	813.27	2.74	195.45	877.58	10.86	–	957.75	20.99	–
19-2-Rnd	50	82	60	1123.94	1212.26	7.86	197.88	1590.74	41.53	–	1594.11	41.83	–
19-2-Mix	50	82	60	1434.45	1517.19	5.77	181.75	1733.38	20.84	–	1865.05	30.02	–
19-2-Pur	50	82	60	1087.98	1120.11	2.95	147.06	1258.88	15.71	–	1328.09	22.07	–
19-3-Rnd	50	103	60	1188.44	1224.59	3.04	227.07	1613.54	35.77	–	1612.66	35.70	–
19-3-Mix	50	103	60	1484.52	1534.46	3.36	191.25	1748.10	17.76	–	1839.38	23.90	–
19-3-Pur	50	103	60	1099.54	1138.03	3.50	132.82	1283.92	16.77	–	1357.02	23.42	–
19-4-Rnd	50	134	60	1213.39	1246.89	2.76	200.90	1633.25	34.60	–	1641.95	35.32	–
19-4-Mix	50	134	60	1480.96	1554.61	4.97	177.80	1794.54	21.17	–	1917.66	29.49	–
19-4-Pur	50	134	60	1138.63	1218.30	7.00	156.21	1313.23	15.33	–	1406.45	23.52	–
19-5-Rnd	50	157	60	1060.30	1079.21	1.78	327.44	1544.18	45.64	–	1544.28	45.65	–
19-5-Mix	50	157	60	1304.16	1334.05	2.29	271.72	1562.40	19.80	–	1761.23	35.05	–
19-5-Pur	50	157	60	976.34	994.85	1.90	205.26	1124.14	15.14	–	1181.94	21.06	–
21-2-Rnd	75	114	120	1767.76	1828.34	3.43	225.93	2338.90	32.31	–	2340.32	32.39	–
21-2-Mix	75	114	120	1799.79	1877.38	4.31	231.19	2172.83	20.73	–	2209.18	22.75	–
21-2-Pur	75	114	120	1409.95	1531.23	8.60	146.17	1747.68	23.95	–	1791.59	27.07	–
21-3-Rnd	75	164	120	1971.75	2053.05	4.12	195.62	2465.74	25.05	–	2464.34	24.98	–
21-3-Mix	75	164	120	1968.64	2134.77	8.44	191.60	2374.29	20.61	–	2407.09	22.27	–
21-3-Pur	75	164	120	1595.27	1668.42	4.59	158.80	1869.91	17.22	–	1907.51	19.57	–
21-4-Rnd	75	168	120	1743.98	1817.47	4.21	207.66	2341.08	34.24	–	2323.37	33.22	–
21-4-Mix	75	168	120	1815.20	1871.71	3.11	209.61	2174.64	19.80	–	2244.16	23.63	–
21-4-Pur	75	168	120	1492.70	1516.72	1.61	155.89	1780.08	19.25	–	1811.23	21.34	–
21-5-Rnd	75	202	120	1630.90	1659.14	1.73	280.62	2237.01	37.16	–	2250.07	37.96	–
21-5-Mix	75	202	120	1728.91	1765.75	2.13	262.50	2067.03	19.56	–	2154.49	24.62	–
21-5-Pur	75	202	120	1321.19	1365.06	3.32	231.73	1607.64	21.68	–	1669.15	26.34	–
25-2-Rnd	100	157	300	2417.89	2559.98	5.88	200.45	3104.05	28.38	–	3106.46	28.48	–
25-2-Mix	100	157	300	2223.29	2380.69	7.08	199.64	2948.81	32.63	–	3081.46	38.60	–
25-2-Pur	100	157	300	2121.95	2254.02	6.22	130.20	2582.53	21.71	–	2646.27	24.71	–
25-3-Rnd	100	212	300	2423.59	2528.47	4.33	206.07	3105.41	28.13	–	3098.09	27.83	–
25-3-Mix	100	212	300	2196.52	2274.95	3.57	242.74	2906.68	32.33	–	3045.32	38.64	–

25-3-Pur	100	212	300	2154.88	2241.20	4.01	165.41	2522.70	17.07	–	2584.63	19.94	–
25-4-Rnd	100	254	300	2471.34	2545.58	3.00	191.98	3088.07	24.95	–	3095.72	25.26	–
25-4-Mix	100	254	300	2186.82	2269.58	3.78	231.34	2886.40	31.99	–	3035.13	38.79	–
25-4-Pur	100	254	300	2166.21	2237.66	3.30	149.58	2552.36	17.83	–	2610.30	20.50	–
25-5-Rnd	100	311	300	2146.53	2185.33	1.81	269.22	2946.79	37.28	–	2929.42	36.47	–
25-5-Mix	100	311	300	1981.35	2013.81	1.64	308.35	2733.71	37.97	–	2889.13	45.82	–
25-5-Pur	100	311	300	1900.61	1932.41	1.67	236.32	2284.15	20.18	–	2392.03	25.86	–
30-2-Rnd	150	225	900	3212.91	3357.04	4.49	192.59	4125.77	28.41	–	4127.65	28.47	–
30-2-Mix	150	225	900	2941.75	3085.51	4.89	246.16	3888.60	32.19	–	3989.23	35.61	–
30-2-Pur	150	225	900	2901.00	3064.66	5.64	136.23	3520.50	21.35	–	3583.79	23.54	–
30-3-Rnd	150	298	900	3319.29	3458.27	4.19	195.97	4218.74	27.10	–	4224.18	27.26	–
30-3-Mix	150	298	900	3085.68	3237.63	4.92	232.76	3999.59	29.62	–	4158.31	34.76	–
30-3-Pur	150	298	900	3141.39	3246.05	3.33	144.04	3656.44	16.40	–	3699.78	17.78	–
30-4-Rnd	150	366	900	3323.61	3432.26	3.27	206.47	4189.11	26.04	–	4181.14	25.80	–
30-4-Mix	150	366	900	3038.70	3126.83	2.90	229.36	3910.82	28.70	–	4021.36	32.34	–
30-4-Pur	150	366	900	3071.03	3195.58	4.06	146.64	3640.62	18.55	–	3656.72	19.07	–
30-5-Rnd	150	433	900	2955.31	3051.51	3.26	275.64	3889.38	31.61	–	3892.39	31.71	–
30-5-Mix	150	433	900	2660.34	2700.66	1.52	322.03	3577.25	34.47	–	3731.25	40.25	–
30-5-Pur	150	433	900	2702.15	2712.68	0.39	224.58	3154.68	16.75	–	3217.27	19.06	–
Average						3.91	198.96		24.78			28.41	

Table 14: Results (travel distances) for different variants of 2L-PDP ("NoRotate" variant, complete results).

Instance				1D	Unrestricted			Simultaneous Packing			Independent Partial Routes		
name	req. n	items m	CPU sec	avg-ttd	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort
09-2-Rnd	25	40	30	704.24	762.44	8.27	136.43	977.09	38.74	–	977.09	38.74	–
09-2-Mix	25	40	30	828.46	951.79	14.89	148.06	1065.49	28.61	–	1107.33	33.66	–
09-2-Pur	25	40	30	827.21	952.69	15.17	99.29	1004.77	21.47	–	1064.96	28.74	–
09-3-Rnd	25	61	30	727.29	752.41	3.45	162.73	963.25	32.44	–	963.25	32.44	–
09-3-Mix	25	61	30	881.69	977.87	10.91	161.75	1086.04	23.18	–	1147.82	30.18	–
09-3-Pur	25	61	30	829.47	953.98	15.01	137.09	1014.14	22.26	–	1070.50	29.06	–
09-4-Rnd	25	63	30	709.10	737.21	3.96	152.77	958.60	35.19	–	958.60	35.19	–
09-4-Mix	25	63	30	847.03	989.50	16.82	87.39	1074.08	26.81	–	1111.18	31.19	–
09-4-Pur	25	63	30	848.14	950.02	12.01	80.60	1006.88	18.72	–	1013.73	19.52	–
09-5-Rnd	25	91	30	706.07	716.79	1.52	189.33	869.40	23.13	–	869.40	23.13	–
09-5-Mix	25	91	30	821.05	823.11	0.25	188.69	957.04	16.56	–	1098.42	33.78	–
09-5-Pur	25	91	30	791.58	815.46	3.02	162.02	877.58	10.86	–	970.54	22.61	–
19-2-Rnd	50	82	60	1123.94	1256.20	11.77	198.62	1651.75	46.96	–	1657.00	47.43	–
19-2-Mix	50	82	60	1434.45	1561.96	8.89	195.70	1782.44	24.26	–	1897.36	32.27	–
19-2-Pur	50	82	60	1087.98	1203.09	10.58	143.66	1348.33	23.93	–	1400.42	28.72	–
19-3-Rnd	50	103	60	1188.44	1258.80	5.92	201.45	1650.19	38.85	–	1651.61	38.97	–
19-3-Mix	50	103	60	1484.52	1586.18	6.85	186.91	1824.13	22.88	–	1870.79	26.02	–
19-3-Pur	50	103	60	1099.54	1198.98	9.04	154.32	1354.76	23.21	–	1437.80	30.76	–
19-4-Rnd	50	134	60	1213.39	1275.84	5.15	186.71	1659.92	36.80	–	1666.77	37.36	–
19-4-Mix	50	134	60	1480.96	1591.69	7.48	173.25	1819.19	22.84	–	1932.04	30.46	–
19-4-Pur	50	134	60	1138.63	1247.50	9.56	126.07	1406.21	23.50	–	1455.46	27.83	–

19-5-Rnd	50	157	60	1060.30	1093.82	3.16	276.44	1563.10	47.42	–	1567.80	47.86	–
19-5-Mix	50	157	60	1304.16	1352.91	3.74	228.17	1578.70	21.05	–	1792.71	37.46	–
19-5-Pur	50	157	60	976.34	1009.30	3.38	189.50	1169.67	19.80	–	1242.20	27.23	–
21-2-Rnd	75	114	120	1767.76	1914.73	8.31	219.04	2427.36	37.31	–	2450.61	38.63	–
21-2-Mix	75	114	120	1799.79	1951.98	8.46	226.31	2251.38	25.09	–	2297.14	27.63	–
21-2-Pur	75	114	120	1409.95	1565.43	11.03	166.22	1840.23	30.52	–	1909.73	35.45	–
21-3-Rnd	75	164	120	1971.75	2089.50	5.97	189.03	2519.17	27.76	–	2521.73	27.89	–
21-3-Mix	75	164	120	1968.64	2177.43	10.61	190.37	2468.48	25.39	–	2514.22	27.71	–
21-3-Pur	75	164	120	1595.27	1712.05	7.32	151.83	1947.55	22.08	–	1979.70	24.10	–
21-4-Rnd	75	168	120	1743.98	1842.86	5.67	186.36	2380.33	36.49	–	2378.51	36.38	–
21-4-Mix	75	168	120	1815.20	1919.86	5.77	192.46	2211.61	21.84	–	2271.51	25.14	–
21-4-Pur	75	168	120	1492.70	1551.43	3.93	134.09	1830.46	22.63	–	1866.97	25.07	–
21-5-Rnd	75	202	120	1630.90	1693.52	3.84	242.49	2266.97	39.00	–	2269.90	39.18	–
21-5-Mix	75	202	120	1728.91	1803.61	4.32	248.00	2107.93	21.92	–	2164.75	25.21	–
21-5-Pur	75	202	120	1321.19	1399.67	5.94	197.78	1638.30	24.00	–	1694.79	28.28	–
25-2-Rnd	100	157	300	2417.89	2678.67	10.79	177.22	3248.80	34.36	–	3260.42	34.85	–
25-2-Mix	100	157	300	2223.29	2516.00	13.17	189.28	3048.13	37.10	–	3215.23	44.62	–
25-2-Pur	100	157	300	2121.95	2384.71	12.38	145.85	2719.28	28.15	–	2789.91	31.48	–
25-3-Rnd	100	212	300	2423.59	2631.36	8.57	209.89	3224.03	33.03	–	3215.21	32.66	–
25-3-Mix	100	212	300	2196.52	2368.35	7.82	212.11	2991.48	36.19	–	3130.54	42.52	–
25-3-Pur	100	212	300	2154.88	2331.26	8.19	155.34	2676.20	24.19	–	2732.23	26.79	–
25-4-Rnd	100	254	300	2471.34	2620.74	6.05	176.39	3165.64	28.09	–	3155.20	27.67	–
25-4-Mix	100	254	300	2186.82	2289.26	4.68	211.66	2956.18	35.18	–	3090.57	41.33	–
25-4-Pur	100	254	300	2166.21	2309.21	6.60	137.48	2667.44	23.14	–	2720.49	25.59	–
25-5-Rnd	100	311	300	2146.53	2210.69	2.99	237.91	2976.51	38.67	–	2983.29	38.98	–
25-5-Mix	100	311	300	1981.35	2052.81	3.61	274.93	2783.52	40.49	–	2908.70	46.80	–
25-5-Pur	100	311	300	1900.61	1963.95	3.33	188.71	2289.93	20.48	–	2419.58	27.31	–
30-2-Rnd	150	225	900	3212.91	3475.64	8.18	175.50	4265.40	32.76	–	4257.73	32.52	–
30-2-Mix	150	225	900	2941.75	3261.25	10.86	206.96	4051.89	37.74	–	4151.40	41.12	–
30-2-Pur	150	225	900	2901.00	3208.83	10.61	131.48	3653.69	25.95	–	3676.27	26.72	–
30-3-Rnd	150	298	900	3319.29	3575.52	7.72	181.68	4320.93	30.18	–	4329.80	30.44	–
30-3-Mix	150	298	900	3085.68	3365.50	9.07	207.16	4138.23	34.11	–	4252.08	37.80	–
30-3-Pur	150	298	900	3141.39	3362.66	7.04	142.79	3825.63	21.78	–	3864.27	23.01	–
30-4-Rnd	150	366	900	3323.61	3525.60	6.08	197.54	4324.26	30.11	–	4291.85	29.13	–
30-4-Mix	150	366	900	3038.70	3219.18	5.94	217.04	4042.58	33.04	–	4161.23	36.94	–
30-4-Pur	150	366	900	3071.03	3289.39	7.11	141.92	3777.89	23.02	–	3799.13	23.71	–
30-5-Rnd	150	433	900	2955.31	3092.13	4.63	247.94	3931.38	33.03	–	3909.83	32.30	–
30-5-Mix	150	433	900	2660.34	2732.56	2.71	299.78	3637.44	36.73	–	3809.21	43.19	–
30-5-Pur	150	433	900	2702.15	2770.56	2.53	183.80	3238.91	19.86	–	3287.70	21.67	–
Average						**7.38**	**182.69**		**28.68**			**32.17**	

Table 15: Total iteration numbers and computing times to find the best solution (“Rotate” variant, complete results).

Instance				Independent Partial Routes			Simultaneous Packing			Ratio iterations
name	req. n	items m	CPU sec	Runtime to best	Runtime to best in %	Total iterations	Runtime to best	Runtime to best in %	Total iterations	
09-2-Rnd	25	40	30	3.14	10.47	677510	4.13	13.77	301744	0.45
09-2-Mix	25	40	30	1.79	5.97	859628	1.87	6.23	343621	0.40
09-2-Pur	25	40	30	6.86	22.87	693085	4.14	13.80	305032	0.44
09-3-Rnd	25	61	30	7.66	25.53	676485	3.46	11.53	296549	0.44
09-3-Mix	25	61	30	3.42	11.40	870142	1.94	6.47	356203	0.41
09-3-Pur	25	61	30	5.64	18.80	735637	7.82	26.07	324160	0.44
09-4-Rnd	25	63	30	3.69	12.30	658966	3.80	12.67	293943	0.45
09-4-Mix	25	63	30	1.50	5.00	858187	5.32	17.73	358785	0.42
09-4-Pur	25	63	30	9.92	33.07	694753	5.42	18.07	315089	0.45
09-5-Rnd	25	91	30	2.50	8.33	660934	0.93	3.10	256483	0.39
09-5-Mix	25	91	30	7.54	25.13	869122	2.54	8.47	335365	0.39
09-5-Pur	25	91	30	4.03	13.43	703858	1.78	5.93	253284	0.36
19-2-Rnd	50	82	60	18.15	30.25	301668	27.04	45.07	121480	0.40
19-2-Mix	50	82	60	29.48	49.13	377188	7.55	12.58	147204	0.39
19-2-Pur	50	82	60	14.77	24.62	276187	21.63	36.05	95434	0.35
19-3-Rnd	50	103	60	25.65	42.75	307649	26.88	44.80	124157	0.40
19-3-Mix	50	103	60	9.89	16.48	365401	10.74	17.90	154127	0.42
19-3-Pur	50	103	60	15.24	25.40	286669	35.86	59.77	97344	0.34
19-4-Rnd	50	134	60	18.80	31.33	291448	16.73	27.88	123299	0.42
19-4-Mix	50	134	60	22.12	36.87	369146	29.31	48.85	150056	0.41
19-4-Pur	50	134	60	18.70	31.17	277825	24.69	41.15	101135	0.36
19-5-Rnd	50	157	60	12.59	20.98	227995	16.64	27.73	98035	0.43
19-5-Mix	50	157	60	9.83	16.38	342502	23.82	39.70	113064	0.33
19-5-Pur	50	157	60	16.21	27.02	232114	10.83	18.05	75663	0.33
21-2-Rnd	75	114	120	55.08	45.90	226599	62.27	51.89	99188	0.44
21-2-Mix	75	114	120	55.68	46.40	269231	69.27	57.73	94611	0.35
21-2-Pur	75	114	120	59.56	49.63	190448	76.24	63.53	62201	0.33
21-3-Rnd	75	164	120	57.04	47.53	265594	58.88	49.07	119277	0.45
21-3-Mix	75	164	120	74.63	62.19	276408	75.12	62.60	113066	0.41
21-3-Pur	75	164	120	79.77	66.48	208837	78.32	65.27	72981	0.35
21-4-Rnd	75	168	120	36.28	30.23	220580	59.64	49.70	89828	0.41
21-4-Mix	75	168	120	72.46	60.38	232932	80.12	66.77	75745	0.33
21-4-Pur	75	168	120	91.05	75.88	160459	80.79	67.33	49184	0.31
21-5-Rnd	75	202	120	80.05	66.71	179722	52.74	43.95	82425	0.46
21-5-Mix	75	202	120	40.37	33.64	205508	92.29	76.91	69756	0.34
21-5-Pur	75	202	120	58.39	48.66	151651	68.93	57.44	38842	0.26
25-2-Rnd	100	157	300	166.90	55.63	360736	211.72	70.57	153021	0.42
25-2-Mix	100	157	300	186.16	62.05	396084	188.72	62.91	150189	0.38
25-2-Pur	100	157	300	179.73	59.91	315825	186.68	62.23	120034	0.38
25-3-Rnd	100	212	300	211.22	70.41	346793	225.58	75.19	151316	0.44
25-3-Mix	100	212	300	128.54	42.85	381004	189.64	63.21	148064	0.39
25-3-Pur	100	212	300	198.51	66.17	313848	181.15	60.38	124748	0.40
25-4-Rnd	100	254	300	163.23	54.41	329294	141.33	47.11	151164	0.46
25-4-Mix	100	254	300	139.60	46.53	372671	184.21	61.40	144303	0.39

25-4-Pur	100	254	300	249.57	83.19	288791	156.02	52.01	116367	0.40
25-5-Rnd	100	311	300	182.83	60.94	243853	221.73	73.91	105327	0.43
25-5-Mix	100	311	300	112.10	37.37	338184	135.55	45.18	122609	0.36
25-5-Pur	100	311	300	214.00	71.33	215152	185.72	61.91	82163	0.38
30-2-Rnd	150	225	900	588.85	65.43	387306	748.72	83.19	169808	0.44
30-2-Mix	150	225	900	639.49	71.05	438919	481.07	53.45	160988	0.37
30-2-Pur	150	225	900	650.04	72.23	342012	754.51	83.83	126217	0.37
30-3-Rnd	150	298	900	761.26	84.58	363763	773.34	85.93	175194	0.48
30-3-Mix	150	298	900	528.54	58.73	428842	666.94	74.10	168398	0.39
30-3-Pur	150	298	900	602.89	66.99	341973	737.58	81.95	146540	0.43
30-4-Rnd	150	366	900	626.66	69.63	340395	821.90	91.32	159951	0.47
30-4-Mix	150	366	900	513.93	57.10	416217	539.58	59.95	164887	0.40
30-4-Pur	150	366	900	763.90	84.88	307661	790.42	87.82	122616	0.40
30-5-Rnd	150	433	900	591.70	65.74	270265	525.86	58.43	127891	0.47
30-5-Mix	150	433	900	357.22	39.69	362924	581.38	64.60	122701	0.34
30-5-Pur	150	433	900	552.52	61.39	224843	736.71	81.86	83153	0.37
Average					**44.78**			**48.63**		**0.40**

Otto von Guericke University Magdeburg
Faculty of Economics and Management
P.O. Box 4120 | 39016 Magdeburg | Germany

Tel.: +49 (0) 3 91 / 67-1 85 84
Fax: +49 (0) 3 91 / 67-1 21 20

www.fww.ovgu.de/femm

ISSN 1615-4274