Received: 28 April 2021 | Accepted: 2 May 2021 | Published on: 4 June 2021

DOI: 10.1002/net.22062

SPECIAL ISSUE ARTICLE

WILEY

Effectiveness of demand and fulfillment control in dynamic fleet management of ride-sharing systems

Jarmo Haferkamp[1] | **Jan Fabian Ehmke**[2]

[1]Management Science Group, Otto von Guericke University Magdeburg, Magdeburg, Germany
[2]Department of Business Decisions and Analytics, University of Vienna, Vienna, Austria

Correspondence
Jarmo Haferkamp, Management Science Group, Otto von Guericke University Magdeburg, Magdeburg, Germany.
Email: jarmo.haferkamp@ovgu.de

Funding information
German Federal Ministry of Transport and Digital Infrastructure, Grant/Award Number: 16AVF2147E

Abstract

In recent years, innovative ride-sharing systems have gained significant attention. In such systems, dynamic fleet management covers demand and fulfillment control to determine which stochastically incoming requests are to be satisfied and how vehicle resources are utilized for their fulfillment, respectively. Demand and fulfillment control can be implemented ranging from straightforward myopic to more sophisticated anticipatory. In this paper, our aim is twofold: (1) we want to classify how policies implement demand and fulfillment control in the related literature on dynamic fleet management; (2) we want to explore the effectiveness of demand and fulfillment control under varying conditions in order to identify benefits and risks for ride-sharing systems. To this end, we define policies that differ in the optimization of demand and/or fulfillment control through the exploitation of either confirmed or complete information. Our experimental results demonstrate that demand and fulfillment control affect the performance and service quality of ride-sharing systems quite differently.

KEYWORDS

anticipation, dial-a-ride problem, dynamic vehicle routing, large neighborhood search, ride-sharing, stochastic requests

1 | INTRODUCTION

Worldwide increasing congestion in urban traffic networks and the associated air pollution have led to a growing interest in innovative shared mobility solutions. Among these are on-demand ride-sharing services like Uber Pool [45], which promise to improve the efficiency of traditional taxi services by bundling travelers on the way from their origin to their destination. This increased level of efficiency allows for lower fares compared to individual taxi services and enables a more convenient travel experience compared to traditional local public transport through smaller transport cabins and direct trips.

Ride-sharing services have emerged in the light of advancing digitization, which allows travelers to submit requests on-demand. The resulting interaction of request acceptance and vehicle routing poses a great challenge on operators, as requests arrive stochastically and decisions have to be made dynamically. In request acceptance, trip requests submitted by travelers—often in expectation of instant confirmation—are processed. Here, it must be ensured that all accepted requests can be fulfilled with the given vehicle resources. Operators may also reject requests due to a lack of resources or in favor of potential future ones. At the same time, vehicle routing addresses the utilization of the fleet to fulfill accepted requests as well as those expected to be accepted in the future. Commonly, accepted requests must be fulfilled at short notice, which means that planning and execution are performed synchronously.

Dynamic fleet management, which comprises request acceptance and vehicle routing to control the demand to be satisfied and its fulfillment, is a key factor for a successful ride-sharing system. To handle the uncertainty caused by the stochastic nature of requests, *demand control* as well as *fulfillment control* can range from simple exploitation of confirmed information only to more sophisticated exploitation of information on the stochastic problem elements. Complex strategies for both demand and fulfillment control are reflected in the decision-making policies proposed in the literature. Given these policies, it remains unclear what extent of demand and fulfillment control is beneficial under which conditions and how it affects the performance of ride-sharing systems. Understanding the effectiveness of demand and fulfillment control is important, both for the systematic development of policies as well as for guiding operators in selecting a policy tailored to the intended service.

Our aim is to investigate the effectiveness of demand and fulfillment control in a ride-sharing system systematically. To this end, first, we define control strategies and classify the policies proposed in the literature accordingly. Secondly, we present policies that vary in the complexity of optimization in demand and/or fulfillment control. Finally, based on a comprehensive computational study, we analyze how these policies affect the performance metrics of a typical urban ride-sharing system as well as the quality of service perceived by travelers.

In our computational study, for the most part, we assume complete information to increase optimization possibilities instead of using truly anticipatory policies. Therefore, we are able to interpret results independent of anticipation capabilities and the quality of information on the stochastic demand. However, concerning the stochastic-dynamic problem, rather upper bounds of effectiveness are investigated than those obtainable by truly anticipatory policies. Nevertheless, it can be assumed that trends in performance differences will be reflected through similar patterns by policies that apply the same strategies toward demand and fulfillment control. Furthermore, to ensure the comparability of the implemented policies, all acceptance and routing decisions are obtained by solving variants of the static dial-a-ride problem (DARP) using a well-known large neighborhood search (LNS).

In summary, we contribute to a better understanding of demand and fulfillment control in dynamic fleet management. To this end, we give an overview of corresponding policies as proposed in the related literature. Moreover, we provide valuable insights into the overall effectiveness of demand and fulfillment control under various conditions of a ride-sharing system.

The paper is organized as follows. Section 2 defines demand and fulfillment control in dynamic fleet management, differentiates control strategies, and provides a classification of the related literature. In Section 3, the dynamic DARP (DDARP) faced by a ride-sharing system is presented and modeled as a Markov decision process. Section 4 covers the framework for investigating the policies as well as the presentation of the LNS. In Section 5, computational experiments are presented including study design and computational results. Finally, Section 6 provides a conclusion and outlines future research directions.

2 | DEMAND AND FULFILLMENT CONTROL

The differentiation between demand and fulfillment control in dynamic fleet management is a key aspect of this paper. We define *demand control* as all dynamic decisions aimed at controlling which trip requests are to be satisfied. We define *fulfillment control* as all dynamic decisions aimed at controlling vehicle resources to fulfill the trip requests to be satisfied. Apart from this logical differentiation, there is a strong interdependence between demand and fulfillment control, since demand control determines the input for fulfillment control and fulfillment control strongly influences the availability of vehicle resources important for demand control. In the following, we detail this differentiation and then classify the policies proposed in the literature.

2.1 | Control strategies

For both demand and fulfillment control, we begin with differentiating whether these are reflected in the decision-making process at all. This distinction is rooted in the few variants of the dynamic vehicle routing problem (DVRP) where either only demand or only fulfillment is dynamically controlled. Secondly, we distinguish how uncertainty is handled within the decision process. Uncertainty is a key challenge in dynamic fleet management caused by the stochastic nature of requests, implying that decisions are made based on incomplete information. We define *anticipatory* decision-making as the consideration of future stochasticity in order to maximize the expected cumulative reward. Anticipatory decision-making can be based on historical data, forecasts, or the distribution of the stochastic elements. In contrast, decision-making that maximizes immediate rewards only based on confirmed information is referred to as *myopic*. In the context of dynamic fleet management, a reward refers, for example, to an accepted request or its monetary compensation. Demand and fulfillment control can be carried out in a myopic or anticipatory manner. This differentiation leads to the following super-ordinate demand and fulfillment control strategies further detailed in the subsections:

- Demand control:
  - *None*: All requests received during the planning period are accepted.

  - *Myopic*: Requests received during the planning period are accepted if sufficient vehicle resources are available and this maximizes the immediate reward.
  - *Anticipatory*: Requests received during the planning period are accepted if sufficient vehicle resources are available and this maximizes the expected cumulative reward.
- Fulfillment control:
  - *None*: Requests accepted during the planning period are scheduled and fulfilled after its completion.
  - *Myopic*: Requests accepted during the planning period are scheduled and fulfilled synchronously within the period.
  - *Anticipatory*: Requests accepted during the planning period are scheduled and fulfilled synchronously within the period, taking into account expected future acceptances.

2.1.1 | Demand control

For the scope of demand control, in this section, we detail the meaning of the strategies referred to as "none," "myopic," and "anticipatory." "None" means that all incoming requests will be accepted. To ensure this, corresponding DDARPs do not consider hard constraints on the quality of service in terms of maximum waiting time for travelers. Instead, service quality becomes part of the objective function in order to achieve a convenient service for all requesting travelers. In practice, however, the demand is usually still controlled indirectly, either on a strategic level through determining a suitable service area, or through inconvenient service offers with long waiting times.

In contrast, myopic and anticipatory demand control accept only a subset of the incoming requests under the objective of maximizing the number of accepted requests or the revenue. This is usually accompanied by strict constraints on the quality of service so that the amount of feasible demand is limited. A key feature of these demand control strategies are customer acceptance mechanisms through which dynamic acceptance decisions are made for each incoming request. The basis for these mechanisms is the so-called *feasibility check*, which ensures that a feasible route plan can be found at any state of the decision process. In the myopic case, demand control is often limited to feasibility checks, as any additional accepted request increases the immediate reward (see, e.g., Coslovich et al. [12]). Enhanced myopic demand control additionally proactively rejects requests if they seem unfavorable at the time of request, for example, in terms of incremental transportation costs (see, e.g., Xiang et al. [48]).

Anticipatory demand control can be implemented in the following ways. The first is through sophisticated customer acceptance mechanisms, which extend the feasibility check to an anticipatory acceptance decision. This ensures that only requests that seem favorable with respect to the expected cumulative reward will be accepted. Whether a request is favorable or not is either reflected in its expected vehicle resource consumption (e.g., Ulmer et al. [46] for a related DVRP) or quantified by a revenue management approach (e.g., Yang and Strauss [49] for a related DVRP). The second way is through the combination of feasibility checks and proactive allocation of vehicle resources by anticipatory routing decisions. Such decisions may concern the relocation of idle vehicles (e.g., Horn [18]) or the incorporation of dummy requests to reflect future expected ones within a scenario-based approach (e.g., Ichoua et al. [21]). As a result, the feasibility check is only successful for favorable requests to which vehicle resources have been allocated. However, the effectiveness of such demand control is strongly dependent on the extent of anticipatory routing decisions and the strictness of service quality constraints.

2.1.2 | Fulfillment control

The classification of fulfillment control into the presented strategies follows the implementation of the routing decisions. Therefore, in case of no control ("none"), no dynamic routing decisions are made. Such policies can be found in the context of reservation systems, where the fulfillment is carried out after a booking process has been completed. Such reservation systems have been investigated in the context of time slot management for attended home deliveries (e.g., Campbell and Savelsbergh [9], Ehmke and Campbell [15]).

When routing decisions are made dynamically as required in on-demand systems, fulfillment control is often based on myopic re-optimization of vehicle route plans in the event of a newly accepted request. This re-optimization can be achieved by applying (meta)-heuristics initially developed to solve a static vehicle routing problem (e.g., in Attanasio et al. [3]), or through newly developed routing algorithms (e.g., in Alonso-Mora et al. [1]).

Anticipatory fulfillment control enhances myopic strategies by considering expected future request acceptances in routing decisions. However, the specific approach can differ greatly in terms of complexity and comprehensiveness. For example, waiting strategies determine at which locations vehicles should wait in order to efficiently accommodate future requests (e.g., Branke et al. [8]). In relocation strategies, vehicles are relocated within the service area for the same purpose (e.g., Horn [18]). The first two examples therefore do not interfere directly with the key task in fulfillment control, namely route planning. In contrast, (multiple)-scenario approaches often plan anticipatory routes with the help of dummy requests (e.g., Ichoua et al. [21]).

TABLE 1 Literature classification

	Demand control		
Fulfillment control	**None**	**Myopic**	**Anticipatory**
None		Campbell and Savelsbergh [10]	Campbell and Savelsbergh [9]
		Ehmke and Campbell [15]	Yang et al. [50]
		Cwioro et al. [13]	Yang and Strauss [49]
			Mackert [25]
Myopic	Dial [14]	**Attanasio et al.** [3]	Ulmer et al. [46]
	Madsen et al. [26]	**Coslovich et al.** [12]	Ulmer et al. [47]
	Ma et al. [24]	**Xiang et al.** [48]	
	Riley et al. [36]	**Beaudry et al.** [4]	
		Berbeglia et al. [6]	
		Berbeglia et al. [7]	
		Hosni et al. [19]	
		Alonso-Mora et al. [1]	
		Lowalekar and Jaillet [22]	
Anticipatory	Mitrović-Minić and Laporte [27]	Branke et al. [8]	**Horn** [18]
	Schilde et al. [40]	Thomas [44]	**Ichoua et al.** [21]
	Hyytiä et al. [20]		**Alonso-Mora et al.** [2]
	Riley et al. [37]		**Shah et al.** [41]
			Yu and Shen [51]
			Pouls et al. [33]

Note: Papers dealing with demand and fulfillment control in a dynamic dial-a-ride problem are written in bold.

Furthermore, for example, approaches of approximate dynamic programming (ADP) provide comprehensive anticipatory fulfillment control for rather small problem instances (e.g., Yu and Shen [51]).

2.2 | Classification of the related literature

In this section, we provide an overview of the related research on dynamic fleet management. In particular, we classify the proposed policies according to the introduced strategies for demand and fulfillment control (see Table 1). To provide a broad overview, we complement the primarily covered papers dealing with DDARP with related ones dealing with customer acceptance mechanisms or DVRP. For a comprehensive literature review on the DARP, we refer to Molenbruch et al. [29] and Ho et al. [17]. For the DVRP, we refer to Psaraftis et al. [34] and Ritzinger et al. [38].

The first studies on dynamic fleet management of a ride-sharing system were conducted by Dial [14] and Madsen et al. [26] in 1995. Dial [14] decomposes the problem into a set of travelling salesman problems, while Madsen et al. [26] suggest an insertion heuristic in order to solve the DDARP. These first contributions focus on myopic fulfillment control without considering demand control. More confirmed policies following this control structure can be found in Ma et al. [24] and Riley et al. [36]. Ma et al. [24] compute solutions for large ride-sharing systems by performing a grid-based service area decomposition. Riley et al. [36] propose a column generation-based policy to satisfy all incoming requests under the objective of minimizing waiting times.

The policy introduced in Riley et al. [36] was later extended in Riley et al. [37] to an anticipatory fulfillment control through a periodic relocation of vehicles in idle mode by means of demand forecasts. More comprehensive anticipatory fulfillment is proposed by Schilde et al. [40] and Hyytiä et al. [20], again without considering demand control. Schilde et al. [40] adapt a multiple-scenario approach, originally introduced by Bent and van Hentenryck [5] for the DVRP, using a variable neighborhood search. Hyytiä et al. [20] propose a theoretical approach combining Markov decision processes and M/M/1 queues to develop an anticipatory policy focusing on fulfillment control for the single-vehicle case.

Anticipatory fulfillment control via waiting time strategies is primarily investigated within the scope of the general DVRP. For example, Mitrović-Minić and Laporte [27] analyze waiting time strategies under the objective of minimizing travel times, while Branke et al. [8] and Thomas [44] maximize the number of accepted requests. Both papers establish demand control through a myopic feasibility check by means of an insertion heuristic. Here, the waiting time strategies themselves are not considered as anticipatory demand control, since stops along the route at which waiting is feasible result from the myopic feasibility checks.

The counterparts to policies that focus only on fulfillment control are customer acceptance mechanisms that focus only on demand control. These mechanisms are primarily examined concerning the problem of managing delivery slots in attended home deliveries. Myopic customer acceptance mechanisms are presented for example in Campbell and Savelsbergh [10], Ehmke and

Campbell [15], and Cwioro et al. [13]. Campbell and Savelsbergh [10] examine financial incentives to encourage customers to choose a delivery time slot that is favorable in terms of myopically planned delivery routes. Ehmke and Campbell [15] compare simple static and dynamic customer acceptance mechanisms under consideration of stochastic travel times. Cwioro et al. [13] propose an adaptive large neighborhood search (ALNS) for the feasibility check to maximize the number of time slots that can be offered. Beyond that, several anticipatory demand control approaches have been considered. Campbell and Savelsbergh [9] adapt the multiple-scenario approach introduced by Bent and van Hentenryck [5] as a customer acceptance mechanism applying an insertion heuristic. Yang and Strauss [49] propose a pricing approach based on ADP using a sophisticated customer choice model developed in Yang et al. [50]. More confirmedly, Mackert [25] approximates opportunity costs using a mixed-integer linear program. Furthermore, the pricing of individual and shared rides for a ride-sharing system is examined in Qiu et al. [35], assuming complete information. Apart from slotting problems, ADP-based customer acceptance mechanisms have also been investigated by Ulmer et al. [46] and extended by Ulmer et al. [47], taking into account fulfillment control. In these two examples, demand control assesses incoming requests with respect to their long-term vehicle resource demand, while fulfillment is myopically controlled using an insertion heuristic.

The papers reviewed above demonstrate that the implementation of demand and fulfillment control within and between DVRPs varies greatly. In the following, we list the variants most relevant for our study, which considers demand and fulfillment control in the scope of a DDARP. Many of those papers propose purely myopic policies. In these cases, well-known solution methods are used to perform a quick feasibility check in demand control as well as to re-optimize route plans in fulfillment control. Such a policy is proposed by Attanasio et al. [3] applying a parallel tabu search (TS) for both tasks, by Coslovich et al. [12] through a two-stage insertion heuristic, by Beaudry et al. [4] through an insertion heuristic for the feasibility check and a TS for re-optimization, and by Berbeglia et al. [6] proposing constraint programming for the feasibility check, which in Berbeglia et al. [7] is extended to a combination of TS and constraint programming.

Some papers propose improved myopic demand control in a DDARP context. The idea is that only cost-effective requests are accepted. This was first discussed for such a problem by Horn [18], yet dismissed due to the potential unfairness toward requests with certain characteristics. Potential discrimination of requests is therefore one aspect that will be examined in our computational study. Xiang et al. [48] and Hosni et al. [19] proposed a policy that proactively rejects requests that seem cost-ineffective myopically. To this end, both papers check feasibility and whether the incremental costs exceed a threshold value. Xiang et al. [48] implement this with an insertion heuristic. Hosni et al. [19] introduce a model-based approach that integrates each incoming request into the incumbent route plan at minimal incremental costs. Furthermore, Alonso-Mora et al. [1] and Lowalekar and Jaillet [22] improve demand control by postponing acceptance decisions until a batch of requests has been received, even if this does not allow for instant trip confirmations. Here, all dynamic decisions are made within a two-stage process. In the first step, a set of potential routes is created, before in the second step an assignment problem is solved to select the routes to be realized under the objective of maximizing the number of accepted requests. While Alonso-Mora et al. [1] determine all feasible combinations of unfulfilled requests in the first step, Lowalekar and Jaillet [22] only consider promising ones with the help of a zone path construction approach.

For the DDARP, several policies have been proposed that implement anticipatory demand control and fulfillment control via routing decisions. A first policy presented by Horn [18] involves the relocation of idle vehicles. More confirmedly, Pouls et al. [33] proposed a policy focusing primarily on anticipatory relocation. In contrast, Ichoua et al. [21], Alonso-Mora et al. [2], Shah et al. [41] and Yu and Shen [51] focus on the anticipatory planning of vehicle routes. Ichoua et al. [21] adapt a multiscenario approach using TS. Alonso-Mora et al. [2] extends Alonso-Mora et al. [1] by incorporating expected future requests via dummy requests. Another extension of Alonso-Mora et al. [1] toward anticipation is proposed by Shah et al. [41], who use ADP for the selection of routes within the allocation problem. ADP is also used by Yu and Shen [51] to solve the DDARP in connection with a decomposition of the problem.

This literature review summarized the different strategies for demand and fulfillment control and how they are implemented through proposed policies. In contrast to the presented literature, with our work, we provide a comparative meta-analysis of demand and fulfillment control to investigate their effectiveness for ride-sharing systems. Moreover, we show under varying system conditions when and how fleet management benefits from which degree of control. With all this, we want to contribute to a better understanding of dynamic fleet management in ride-sharing systems and encourage the systematic development and selection of policies concerning their intended effectiveness.

3 | PROBLEM FORMULATION

In this section, we define the components of the DDARP under consideration. Then, we model the stochastic and dynamic problem as a Markov decision process, enabling demand and fulfillment control in a ride-sharing system through dynamic acceptance and routing decisions.

3.1 | Problem components

Let $\mathcal{L}$ be a set of locations in the service area of a ride-sharing system. For each location $l \in \mathcal{L}$, it is assumed that a (deterministic) service time p_l for travelers getting on and off a vehicle is known, as well as for all pairs of locations $(i,j) \in \mathcal{L}$, a (deterministic) travel time of $c_{i,j}$ is defined. The considered ride-sharing system faces a demand represented by trip requests $r \in \mathcal{R}$. Each request is characterized by its receiving time t_r, its origin $o_r \in \mathcal{L}$, its destination $d_r \in \mathcal{L}$, as well as its fulfillment time window $[b_r, e_r]$, which defines the earliest pick-up time b_r and latest drop-off time e_r. We assume that the earliest pick-up time b_r corresponds to the receiving time of the request t_r. This means that travelers must be ready for departure at the time when they pose their request, which excludes prebookings. The latest drop-off time e_r is defined by addition of earliest pickup time b_r, direct travel time c_{o_r,d_r}, and a parameter α, which defines the maximum arrival delay tolerated by travelers. Arrival delays arise from waiting time to be picked up as well as detours caused through the bundling of requests. Detours include both additional travel time to reach the origin or destination of other travelers and the service time required by them for getting on or off the vehicle. To satisfy the demand, a fleet of identical vehicles $\mathcal{V}$ is available. We assume that the capacity of a vehicle is not constraining, that is, passenger seats are never fully occupied due to tight time windows for the request fulfillment.

3.2 | Markov decision process

The considered decision process consists of a series of decision epochs $k \in \mathcal{K}$, covering a temporally limited planning period of a DDARP. At the beginning of the planning period, the service is in an initial state s_0. For this state, we assume that the vehicles $v \in \mathcal{V}$ are waiting in idle mode at an initial location $l_v \in \mathcal{L}$. Furthermore, it is assumed that in the initial state s_0 no trips are waiting for fulfillment. Accordingly, a degree of dynamics as defined by Lund et al. [23] as the ratio of the demand stochastically received to the total demand, of 100% is assumed. Each decision epoch $k \in \mathcal{K}$ is triggered by a stochastically incoming request $r_k \in \mathcal{R}$ leading to a predecision state s_k. The predecision state reflects all decision-relevant characteristics such as the activities of the vehicles and pending requests. Formally, the predecision state s_k is defined by the time t_r at which the service operator has received the new request r_k. Furthermore, it contains the state of the resources described through the tuple $(l_k^v, \mathcal{O}_k^v | \forall v \in V)$, where $l_k^v \in \mathcal{L}$ specifies the current vehicle locations and $\mathcal{O}_k^v \subset \mathcal{R}$ for each vehicle the set of accepted requests whose travelers are currently being transported. Finally, it represents the demand described through the tuple $(r_k, \mathcal{U}_k)$, where r_k refers to the new request and $\mathcal{U}_k \subset \mathcal{R}$ to the set of accepted requests whose travelers still have to be picked up. These three parts result in the state definition $s_k = (t_r, (l_k^v, \mathcal{O}_k^v | \forall v \in V), (r_k, \mathcal{U}_k))$.

Based on the predecision state s_k, an action $A^\pi(s_k)$ is derived from a policy $\pi \in \Pi$. A policy $\pi \in \Pi$ thus defines for each predecision state s_k all decisions to be taken and can thus be considered as a solution approach to the stochastic-dynamic problem. Here, an action consists of two hierarchically dependent decisions. The first decision is whether to accept or reject the new request r_k. This acceptance decision is represented by the binary decision variable $x_k \in \{0, 1\}$, where $x_k = 1$ represents acceptance and $x_k = 0$ represents the rejection of a request. The second decision is the selection of a feasible route plan, defining the utilization of all vehicles $v \in \mathcal{V}$ until the next decision epoch. A route plan is considered feasible if all accepted requests have been assigned to a vehicle subject to the following constraints:

(i) For all pending accepted requests $r \in \mathcal{U}_k$ and the new request r_k, in case of $x_k = 1$, the pick-up at origin o_r is planned before the drop-off at destination d_r for the same vehicle $v \in \mathcal{V}$.
(ii) For all currently executed requests $r \in \mathcal{O}_k^v$, the drop-off at destination d_r is planned for the same vehicle $v \in \mathcal{V}$.
(iii) For all origins, the planned pick-up z_o is later or at the same time as the corresponding earliest pick-up time b_r.
(iv) For all destinations, the planned drop-off z_d is earlier or at the same time as the corresponding latest drop-off time e_r.

Let $y_k \in \mathcal{F}_x$ be the routing decision variable, with $\mathcal{F}_x$ as a finite set of all feasible route plans under consideration of decision x_k. Such a set could for example be determined heuristically by adapting a solution method developed for a static vehicle routing problem. The acceptance decision x_k requires that the set of all route plans $\mathcal{F}_x$ must not be empty. The execution of action $A^\pi(s_k)$ leads to a deterministic transition from the predecision state s_k to a postdecision state $s_k^a = (y_k)$. This state consists of the selected feasible route plan y_k, which serves for the routing of the vehicles until the next decision epoch $k+1$. This is triggered by the stochastic transition W_{k+1}, which reflects that the operator has received the next request $r_{k+1} \in \mathcal{R}$.

Let B_k be the partial reward function for one decision epoch $k \in K$ and let the value of B_k be equal to the acceptance decision x_k, so that the cumulative reward $v^\pi(s_0)$ corresponds to the number of accepted requests. The objective is to find an optimal policy $\pi^* \in \Pi$ that maximizes the expected cumulative reward $v^\pi(s_0) = \max^\pi \mathbb{E}\{\sum_{k=0}^{K} B_k(s_k, A^\pi(s_k), W_{k+1}) | s_0\}$ over all decision epochs $k \in \mathcal{K}$. Having formally introduced the stochastic-dynamic problem under consideration, in the next section, we present policies that exploit only confirmed or complete information to solve the problem with varying degrees of optimization in demand and/or fulfillment control.

TABLE 2 Investigated policies

	Demand control	
Fulfillment control	**Inspired by myopic strategies**	**Inspired by anticipatory strategies**
Inspired by myopic strategies	*Basic Control*	*Advanced Demand Control*
Inspired by anticipatory strategies	*Advanced Fulfillment Control*	*Advanced Control*

4 | EVALUATION FRAMEWORK

In this section, we describe our evaluation framework for investigating the effectiveness of demand and fulfillment control within dynamic fleet management of a ride-sharing system. Using the control strategies discussed in Chapter 2, we detail the implemented policies in Section 4.1 and discuss an established LNS that we use to realize them in Section 4.2.

4.1 | Implementation of policies

From the control strategies defined in Section 2.1, we derive possible combinations of demand and fulfillment control to define policies for dynamic fleet management. These policies differ in optimization capabilities in demand and/or fulfillment control through the exploitation of confirmed information only or complete information. Table 2 summarizes the policies with regard to the related control strategy.

Basic Control refers to policies employing purely myopic demand and fulfillment control. They are implemented through a feasibility check for demand control and re-optimization of routes for fulfillment control. In particular, the feasibility check for the acceptance decision x_k is made by an insertion heuristic, checking whether an incoming request $r_k \in \mathcal{R}$ can be inserted into the incumbent route plan y_{k-1}, where y_0 refers to the initial empty route plan. The route plan obtained is then re-optimized in the scope of the routing decision y_k. For this purpose, a static DARP is solved considering all accepted, not yet fulfilled requests under the objective of minimizing total travel time. Note that for both acceptance and routing decisions, already fulfilled requests as well as locations approached by a vehicle will not be rescheduled. This means that vehicles are not tracked along the path between two locations $i, j \in \mathcal{L}$, which reduces the rescheduling opportunities but also the computational effort related to locating vehicles. Moreover, it allows drivers and travelers to be reliably informed about the next stop, avoiding frequent diversions of vehicles.

From an operator perspective, *Basic Control* could be advantageous in case of highly uncertain conditions, where the inclusion of additional information in optimization does not pay off. Moreover, *Basic Control* does not require any sophisticated technological and computational resources. The key argument against *Basic Control* is the high risk of insufficiently informed decision-making both for demand and fulfillment control.

Advanced Demand Control aims at improving the performance of a ride-sharing system through the acceptance of favorable requests in terms of vehicle resource occupancy (e.g., requests which can be bundled more easily). It is inspired by policies that implement anticipatory demand control through sophisticated customer acceptance mechanisms while fulfillment is controlled myopically. Therefore, complete information is exploited to enable enhanced acceptance decisions x_k, while routing decisions y_k are made based only on confirmed information.

In particular, the acceptance decision x_k is made for each incoming request $r_k \in \mathcal{R}$ in a two-step procedure. First, a feasibility check is carried out by an insertion heuristic (as in *Basic Control*). If the feasibility check has been successful, the favorability of the request is investigated in the second step. To identify favorable requests, a static team orienteering problem (TOP) with equal scores for each considered request is solved. The TOP is a well-known variant of the static vehicle routing problem, in which only the most cost-efficient locations are visited. The objective is to find the optimal set of visited locations which maximizes the operator's benefit [11]. As input for the TOP serves all requests of the incumbent route plan y_{k-1}, the current request r_k as well as all future requests. All requests have equal scores, representing that the route plan found maximizes the number of integrated requests. All requests of the incumbent route plan y_{k-1} must be covered to identify favorable requests among current and future requests. In the end, a request r_k is accepted if it is contained in the best route plan found. After the acceptance decision has been made, a new route plan y_k is determined by solving a static DARP without taking future requests into account, following re-optimization based fulfillment control in *Basic Control*.

In summary, with *Advanced Demand Control*, a ride-sharing system operates more efficiently through the controlled selection of the requests to be satisfied. However, since we assume basic fulfillment control, vehicle resources may be utilized in an unfavorable way making beneficial demand control much more challenging. Furthermore, there are risks associated with a selective demand control such as incomprehensible rejections as well as rejections perceived as proactive, leading to dissatisfied travelers. Moreover, the continuous rejection of certain requests identified as unfavorable may prevent such trips from being requested, regardless of whether their assessment might change over time.

Advanced Fulfillment Control improves request fulfillment by considering all future request acceptances in conjunction with basic demand control. It is inspired by policies that implement anticipatory approaches for fulfillment control only.

In particular, complete information is exploited to obtain a favorable route plan in advance with respect to a feasibility check based demand control. To this end, request acceptance decisions are simulated for each future request $r \in \mathcal{R}$ in order of appearance through an insertion heuristic. However, the fulfillment of accepted requests is *not* simulated, so that the incumbent route plan can be changed flexibly throughout these checks. Once all decisions on request acceptance have been simulated, a route plan is created from the obtained acceptances as a blueprint for fulfillment control.

Summarized, *Advanced Fulfillment Control* enables an optimized fulfillment of the accepted requests without changing the concept of demand control. The acceptance of a request therefore depends primarily on the time a request is posed and not, like in case of *Advanced Demand Control*, on its characteristics. However, *Advanced Fulfillment Control* may even reinforce the drawbacks of such an basic demand control by enabling the acceptance of more demanding requests through improved vehicle routing.

Finally, ***Advanced Control*** exploits complete information on future demand for both demand and fulfillment control allowing all dynamic decisions to be made in advance. It is inspired by policies in which demand and fulfillment are controlled through anticipatory approaches. The exploitation of complete information is done by solving a static TOP with the same score for each request $r \in \mathcal{R}$. This results in a route plan that maximizes the number of integrated requests so that the requests to be accepted and the routes to be taken can be selected accordingly. It therefore naturally outperforms the other three policies.

4.2 | Large neighborhood search

In the following, we describe how the different policies are implemented based on an LNS. We apply the same heuristic to all occurring DARPs to ensure the comparability of policies within computer experiments. The developed LNS is based on the ALNS proposed by Ropke and Pisinger [39]. It was chosen because it has been applied over years to a variety of complex vehicle routing problems and has achieved consistently good results in relatively short run times.

4.2.1 | Overview

The basic idea of an LNS is to destroy and repair solutions iteratively [31]. For the problem at hand, a solution w is represented by a route plan n_w and a set of unplanned requests $m_w \in \mathcal{R}$, whose fulfillment is not yet considered in route plan n_w. A route plan n_w consists of a plan for each vehicle $v \in \mathcal{V}$, which specifies the sequence of the locations $l \in \mathcal{L}$ to be visited as well as their planned arrival times z_l^v. The LNS aims to maximize the number of request fulfillments $|n_w|$ and/or to minimize the required total travel time $c(n_w)$. Algorithm 1 presents the pseudocode of our LNS implementation.

The search is initialized with a solution w_0 as input, which is saved as incumbent solution w and best known solution w_{best} (lines 2 and 3). Next, the iterative search for a superior solution is performed until a termination criterion is met. As a termination criterion, the maximum number of iterations β is defined as well as further criteria depending on the respective purpose of the search. Each iteration of the LNS begins with the creation of a new solution (lines 5–7). For this purpose, the incumbent solution w is saved as the basis of the new solution w_{new}. Afterwards, w_{new} is destroyed through an operator that moves between γ_1 and

Algorithm 1. Large neighborhood search

1 **Function** LNS(w_0)
2   $w = w_0$
3   $w_{\text{best}} = w_0$
4   **while** *termination criterion is not met* **do**
5     $w_{\text{new}} = w$
6   remove requests from $n_{w_{\text{new}}}$ to $m_{w_{\text{new}}}$
7   insert requests from $m_{w_{\text{new}}}$ into $n_{w_{\text{new}}}$
8   **if** *(w_{new} is accepted)* **then**
9       $w = w_{\text{new}}$
10   **if** *(w_{new} is an improvement to w_{best})* **then**
11         $w_{\text{best}} = w_{\text{new}}$
12   **end**
13   **end**
14   **end**
15   **return** w_{best}

γ_2 percent of the requests from the route plan $n_{w_{\text{new}}}$ to the set of unplanned requests $m_{w_{\text{new}}}$. If in the dynamic environment the origin o_r has been visited already, the corresponding destination d_r is no longer removable. The exact number of requests to be removed is determined in each iteration by a random value q_1 with $\{q_1 \in \mathbb{N} \mid (\gamma_1 \times |n_{w_{\text{new}}}|) \leq q_1 \leq (\gamma_2 \times |n_{w_{\text{new}}}|)\}$. In the next step, a repair operator inserts as many requests from the set of unplanned requests $m_{w_{\text{new}}}$ into the route plan $n_{w_{\text{new}}}$ as feasible. For both destroy and repair operators, in contrast to a classical ALNS, the particular operator is selected randomly for each iteration. This is a consequence of the implementation of the LNS in a dynamic environment, where multiple searches are performed over a few iterations so that automatic adaptation of the operator selection during the search is neither feasible nor advantageous.

Removal operators correspond to those used in Ropke and Pisinger [39]. We summarize them as follows:

Random-removal: This operator randomly selects the requests to be removed and thus provides a maximum diversification in terms of the set of selected requests.

Worst-removal: The aim of this operator is to remove requests that are not placed well. For this purpose, all requests of a route plan are sorted in descending order in a list according to the travel time that could be saved if the request was removed. In order to avoid the repeated removal of similar sets of requests, “noise” is applied when selecting a request for removal. Following Ropke and Pisinger [39], we use the formula $q_2^{\delta_1} \times |list|$ to determine the list position of the next request to be removed. In this formula, q_2 stands for a random value with $\{q_2 \in \mathbb{Q} | 0 \leq q_2 \leq 1\}$ and δ_1 for the parameter that controls the degree of noise.

Shaw-removal: Originally introduced by Shaw [42], this operator removes similar requests, since they can be shuffled around more easily so that improved route plans can be found more likely. In particular, first, a request is randomly selected. All other requests are then sorted in ascending order according to their similarity to the selected request and removed corresponding to the sorting. The similarity between two requests r_1 and r_2 is calculated by the distances between origins $c_{a_{r_1},a_{r_2}}$ and destinations $c_{d_{r_1},d_{r_2}}$ as well as between their planned arrival times $\Delta(z_{a_{r_1}}, z_{a_{r_2}}) + \Delta(z_{d_{r_2}}, z_{d_{r_2}})$. Before the geographical and temporal values are added up, they are min-max normalized.

For the subsequent insertion of the removed requests, there is a wide range of operators. We discuss only those operators that turned out promising in previous tests, one with and one without noise:

Regret-2-insertion: The regret-insertion heuristic was first proposed by Potvin and Rousseau [32] for the vehicle routing problem with time windows. The idea is to insert requests where the regret would be largest if the best found insertion option was no longer feasible. An insertion option comprises a position in a route for the origin and the destination of a request. For the regret-2 variant, the regret is calculated from the difference between the most and the second most cost-effective feasible insertion option. The costs correspond to the additional travel time which would result from the request being inserted. In case that only one feasible insertion option can be found, the difference to the maximum integer value is calculated instead. For each selection of the next request to be inserted in the route plan, the regret value of each unplanned request $r \in m_{w_{\text{new}}}$ is calculated and sorted in descending order. For the operator without noise, the request with the highest regret value is inserted into the most cost-effective feasible position. For the operator with noise, the selection of the next request to be inserted is made in the same way as described for the worst-removal operator. The degree of noise is controlled in this case by the parameter δ_2.

A new generated solution w_{new} is accepted if the number of planned requests $|n_{w_{\text{new}}}|$ remains equal or increases relative to the incumbent solution w (see line 8). Since mostly fully utilized services are investigated, which often show limited routing flexibility, this acceptance criterion has the advantage of allowing a maximum diversification with respect to the overall travel time and prevents deterioration of the number of planned requests. After accepting and saving s_{new} as incumbent solution w, it is checked whether this solution is superior to the best-known solution w_{best} (lines 10–12). This is the case if the number of planned requests $|n_{w_{\text{new}}}|$ is higher or remains equal with a shorter total travel time $c(n_{w_{\text{new}}})$. After evaluating the new solution w_{new}, the next iteration is performed until the search is terminated, and the best known solution w_{best} is returned (line 15).

4.2.2 | Execution

In the following, we briefly describe how the LNS is applied to execute the before outlined acceptance and routing decisions of the four policies under consideration.

Basic Control: In each decision epoch $k \in \mathcal{K}$, the LNS is first executed to perform the feasibility check-based acceptance decision x_k. To this end, the input set of unplanned requests m_{w_0} is represented by request r_k, while input route plan n_{w_0} corresponds to an empty plan in case of $k = 0$ and route plan y_{k-1} otherwise, yet updated with respect to the time of request t_k. Through the update, input route plan n_{w_0} covers only stops at locations whose planned arrival time z_l^v plus service time p_l is greater or equal to the time of request t_k. Furthermore, the first location of each plan contained in n_{w_0} represents the current respectively next location of a vehicle and will not be rescheduled. Based on this input, the LNS searches for a solution w_{new} that covers all requests in the route plan $n_{w_{\text{new}}}$. The search is terminated when either such a solution has been found ($x_k = 1$) or a maximum of β iterations has been performed ($x_k = 0$). Note that in case of an unsuccessful feasibility check, the returned solution is discarded, while the updated routing decision y_{k-1} serves as routing decision y_k. In case of a successful feasibility check, the routing decision y_k is determined by re-optimization of the found solution n under the objective of minimizing the total travel time $c(n_w)$ in β iterations.

Advanced Demand Control: This policy requires in each decision epoch $k \in \mathcal{K}$ a feasibility check for the acceptance decision x_k as well as a re-optimization for the routing decision y_k. It generally corresponds to the procedure of *Basic Control*. However, after each successful feasibility check follows the additional favorability check of the acceptance decision x_k. To solve the corresponding TOP, the initial solution w_0 consists of the same route plan as in the feasibility check. The set of unplanned requests m_{w_0} contains, besides the new request r_k, all trips to be requested in the following decision epochs $k+1, k+2, \dots, k+n$. Based on this input, the LNS is executed to maximize the number of planned requests $|n_w|$. For the acceptance of a new solution w_{new} as best solution w_{best}, an additional criterion is applied, which evaluates if all requests contained in the initial route plan n_{w_0} are as well contained in $n_{w_{\text{new}}}$. The search terminates after either finding a solution w_{new} that contains all requests considered in the search or after β iterations have been performed. Once the search has been terminated, it is examined whether the candidate request r_k is contained in the returned route plan $n_{w_{\text{best}}}$, which represents that the request has passed the favorability check.

Advanced Fulfillment Control: In this case, the LNS is primarily executed as a feasibility check to obtain the future request acceptances. The input of this feasibility check differs from that presented for *Basic Control* by omitting time-related updates of the incumbent route plan so that it remains flexible throughout all checks. After completion of the last feasibility check, the LNS is executed to determine the route plan thought of as a blueprint for the decision process. To this end, the solution returned by the last successful feasibility check is optimized under the objective of minimizing the total travel time $c(n_w)$ in β iterations.

Advanced Control: Here, the LNS is executed to solve a TOP again to obtain a route plan thought of as a blueprint for the decision process. The initial solution w_0 consists of an empty route plan n_{w_0}, and the set of unplanned requests m_{w_0} includes all requests $r \in R$. The solution w is then optimized in β iterations with respect to the number of planned requests $|n_w|$ and the total travel time $c(n_w)$.

5 | COMPUTATIONAL EXPERIMENTS

In this section, we analyze the impact of the presented policies on the effectiveness of demand and fulfillment control with respect to the performance of the ride-sharing system. We introduce our instances and present the results of the computational study. The description of the parameter tuning of the LNS is given in Appendix.

From the computational results, we first analyze the performance regarding the achieved solution quality expressed as acceptance rate, defined by the number of accepted requests divided by the number of received requests. Secondly, further metrics that describe the operational performance of the ride-sharing system are discussed. This provides insights into the nature of such systems and contributes to a better understanding of the context-related effectiveness of demand and fulfillment control. Thirdly, we investigate the effect on the service quality perceived by travelers through a detailed trip-specific evaluation. Last, we analyze how acceptance rates change when information becomes incomplete.

5.1 | Experimental design

Our case study is based on taxi trip data collected in the urban area of New York City, USA. This dataset is provided by the City of New York and contains a total of 165,114,361 million trips fulfilled by the Yellow Cab taxi fleet in the year 2014 [30]. Each record contains the start and end time of the trip, the distance traveled as well as the origin and destination locations in terms of geographical coordinates. Figure 1 shows the temporal distributions of the trips. In order to simplify the data handling and to ensure consistent trip patterns, we only include weekday trips from January 2014 that operate in the evening peak (as indicated in Figure 1 between 5:30 p.m. and 8:30 p.m.) in the area of Manhattan. Furthermore, only trips with a distance greater than zero are considered.

Given the taxi trip data, we derive the characteristics of our ride-sharing system as follows. First, potential initial vehicle locations are determined. For this purpose, 40 locations were randomly sampled from the set of locations where a trip ends at 5:30 p.m. Second, potential trip requests including origins and destinations are defined. To this end, of all included trips, 180 were randomly sampled. Thus, we assume one incoming request per minute on average. A constant set of trip requests is used in all experiments to enable trip-specific evaluations. The selected locations are visualized in Figure 2, indicating that there is a centrally located area in Manhattan with a higher demand density. Next, free-flow travel times between all locations were computed using the GraphHopper routing engine [16]. Free flow travel times are multiplied by factor ϵ to provide a simple approximation to the real travel times during peak hours.

In summary, we create 110 problem instances as follows: 10 instances are used for the parameter tuning of the LNS, and 100 for our computational study. These instances differ in the receiving times of each request as well as in the initial vehicle locations. Moreover, a baseline scenario is defined for all instances as follows: a fleet of 10 vehicles, a planning period from 5:30 to 8:30 p.m. (180 min), a travel time factor $\epsilon = 3$, and a maximum arrival delay for each request of 15 min. We vary the baseline scenario as follows. First, we vary the fleet size to analyze varying resource-demand ratios. Second, we analyze

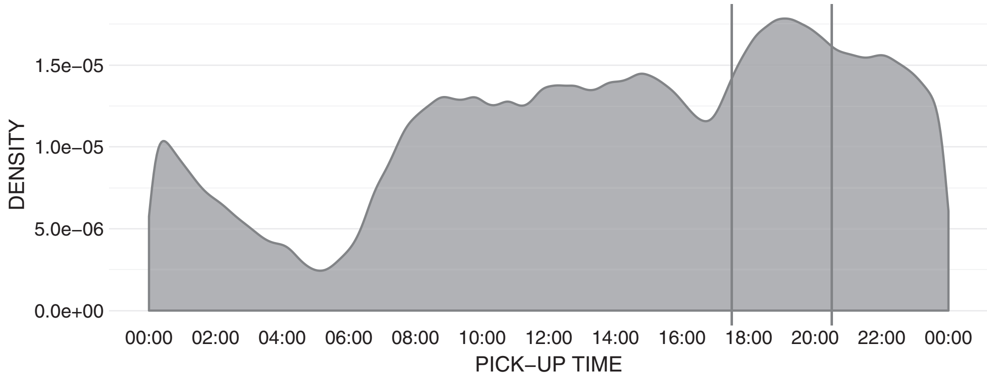

FIGURE 1 Pick-up time distribution

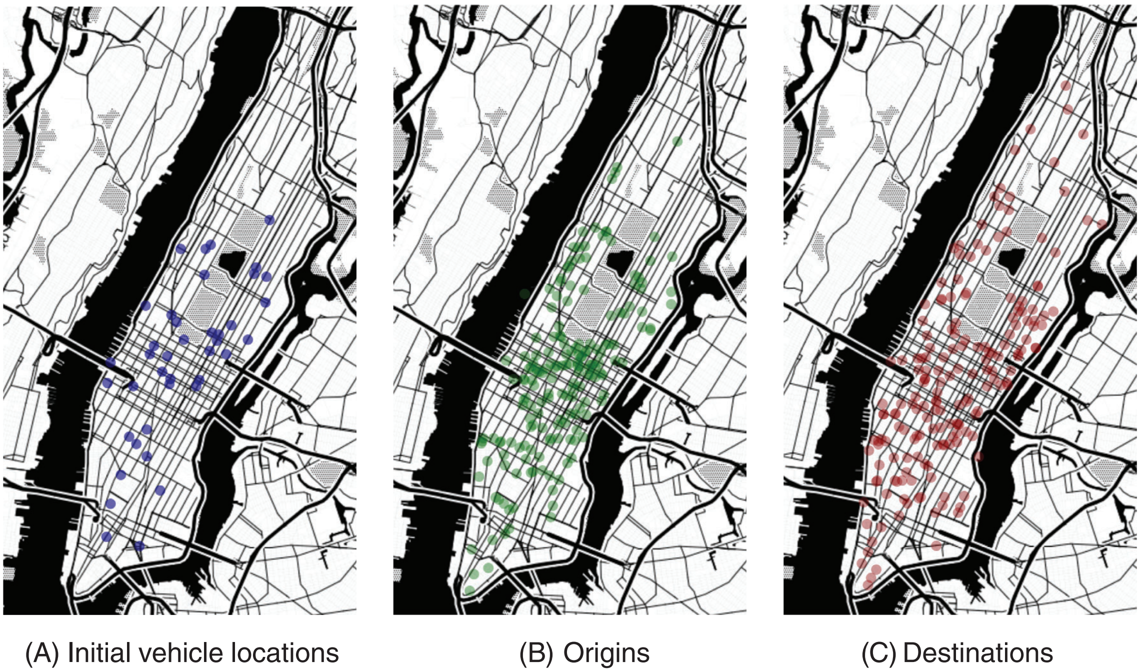

FIGURE 2 Location distributions. (A) Initial vehicle locations; (B) Origins; (C) Destinations

TABLE 3 Values for the sensitivity analyses

Sensitivity analyses	Varying characteristic	Values				
Resource Demand Ratio	Fleet size	2	6	**10**	14	18
Temporal Demand Density	Planning period	36 min	108 min	**180 min**	252 min	324 min
Geographical Demand Density	Factor on travel time	0.6	1.8	**3**	4.2	5.4
Fulfillment Time Window	Maximum arrival delay	3 min	9 min	**15 min**	21 min	27 min

The values are highlighted on the basis of which one is varied in each of the analyses.

the impact of temporally varying demand density. To this end, the length of the planning period is varied, and receiving times of requests are adjusted to the corresponding time frame under investigation, whereby 7:00 p.m. always marks the middle of the planning period. Third, we analyze geographically varying demand densities by adjusting the travel time factor. Fourth, we examine the impact of the fulfillment time window by varying the allowed maximum arrival delay.

For each analysis, four variations of the base value are considered, representing a decrease of 40% and 80% as well as an increase of 40% and 80% of its parameters (see Table 3). With these parameter intervals, we can cover a wide range of possible objective function values and at the same time create insights into where and when the effectiveness of the considered policies is changing. However, to keep the computational effort manageable, only one parameter is varied at a time, while all others keep the value highlighted in Table 3.

In the next section, we will discuss the results of all four sensitivity analyses concerning their impact on acceptance rates. In the subsequent sections, we focus on *Resource Demand Ratio*, while detailed results for the other sensitivity analyses can be found in the Appendix, as the results are structurally similar.

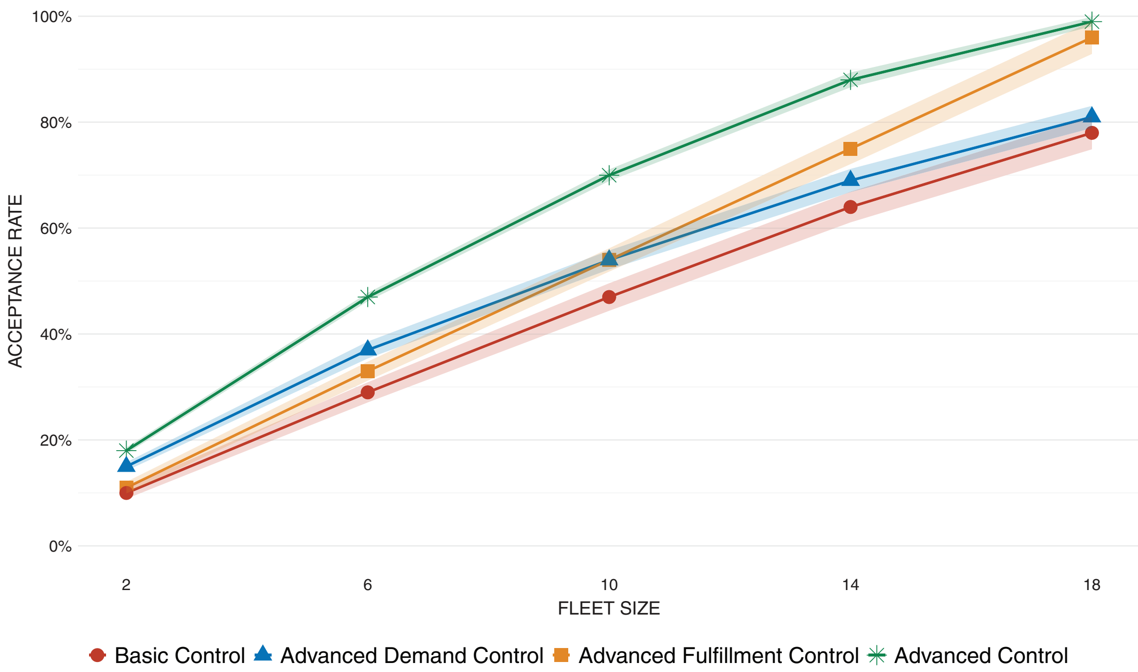

FIGURE 3 *Resource Demand Ratio*: Average acceptance rates with their standard deviations

5.2 | Analysis of acceptance rates

We begin by analyzing the effectiveness of the presented policies with respect to acceptance rates. We particularly analyze the value of more advanced optimization in demand and fulfillment control. Overall results are presented in Figure 3, which shows the acceptance rate on the Y-axis and the fleet size on the X-axis. The acceptance rates are calculated based on the 100 instances solved five times with the varying fleet sizes for each of the four policies. The differently shaped points represent the numeric results and the trend is highlighted by connecting lines. Additionally, the standard deviations of the average acceptance rates are illustrated by a lighter color range around the lines.

Generally, with increasing fleet size, achievable acceptance rates increase as well. As expected, *Basic Control* leads to the smallest acceptance rates, while *Advanced Control* creates the best acceptance rates with an increase about $10\% - 20\%$ compared to *Basic Control*. Interestingly, for smaller fleet sizes, *Advanced Demand Control* yields better results, while for larger fleet sizes, *Advanced Fulfillment Control* can create significantly higher acceptance rates. The standard deviations increase with increasing fleet sizes. They are negligible for *Advanced Control*. It can be concluded that the highest potential lies in the advanced control of both demand and fulfillment, regardless of the resource demand ratio. However, the contribution to this potential shifts from demand to fulfillment control with an increasing acceptance rate.

We now analyze the results of the further sensitivity analyses (see Figure 4). We begin with *(A) Temporal Demand Density*, where we manipulate the demand through temporal variation of the planning period. Generally, results are similar to those obtained for the *Resource Demand Ratio* analysis. For the same fleet size, a relatively larger planning period allows accommodating more requests, with a high benefit of advanced fulfillment control for a large temporal spread of requests and a high benefit of advanced demand control for a small temporal spread of requests. For *(B) Geographical Demand Density*, instead of the time of the planning period, the travel time factor ϵ is used to vary the geographical density of the service area. As expected, when the relative travel times become larger and the area of operation becomes more "stretched" out, the acceptance rates decrease. The acceptance rate of *Advanced Control* is about 20% higher than for *Basic Control*. Advanced control at either demand or fulfillment can improve this by about 5% only. Here, a high geographical density diminishes the benefits of advanced demand control and increases those of advanced fulfillment control. However, when the geographical density decreases, unfavorable requests from remote regions may automatically be infeasible to fulfill. Finally, we analyze for *(C) Fulfillment Time Window* how the variation of the maximum delay, consisting of waiting time and detour, influences the effectiveness of demand and fulfillment control. As expected, acceptance rates increase for all policies with an increased maximum delay. However, the gap between *Basic Control and Advanced Control* is very large for small maximum delays. In contrast, *Advanced Fulfillment Control* yields quite stable results for all maximum arrival delays. The benefit from enhanced demand control is higher when the maximum delay is higher. In contrast, the benefit from enhanced fulfillment control is higher when the maximum delay value is lower.

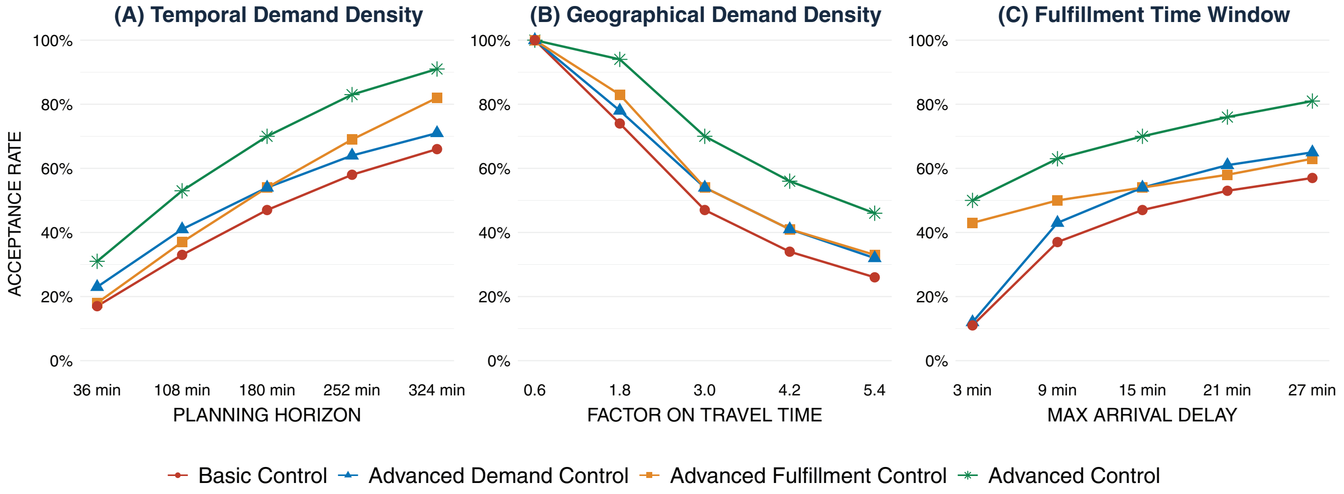

FIGURE 4 Average acceptance rates per sensitivity analyses. (A) Temporal demand density; (B) Geographical demand density; (C) Fulfillment time window

The above findings demonstrate that the effectiveness of demand and fulfillment control depends highly on the system characteristics. However, particularly advanced control of demand and fulfillment shows great potential to increase the acceptance rate of a ride-sharing system. Furthermore, it becomes clear that the potential for demand and fulfillment control differs in response to the characteristics of the system under consideration. The value of advanced demand control is particularly high when (1) insufficient resources (due to small fleet size or dense temporal demand) require a significant proportion of requests to be rejected, and (2) when a sufficiently large and heterogeneous pool of potentially acceptable demand (due to moderate geographic demand density and sufficiently wide fulfillment time windows) enables the selection of more favorable requests. For advanced fulfillment control, the analysis of different fulfillment time windows demonstrates its importance when offering immediate pick-up times, while the others highlight the dependency on a sufficiently high acceptance rate. Hence, with only a few accepted requests, the trips to be fulfilled are so unfavorable that an increase in performance through advanced fulfillment control alone is barely achievable. Overall, the results imply that the potential of policies focusing on an advanced demand *or* fulfillment control only vary greatly depending on the nature of the ride-sharing system.

5.3 | Analysis of operational performance

The aim of this subsection is to gain further insights into how demand and fulfillment control impact further performance metrics of a ride-sharing system. The following metrics are considered:

- The average travel time per fulfilled request, defined as the total travel time divided by the total number of fulfilled requests.
- The pooling rate, which measures the percentage of travelers who shared a part of their ride with at least one other traveler.
- The percentage share of each vehicle mode, defined by the total time all vehicles have spent in the mode, is divided by the total time spent by the entire fleet. The considered modes are:
    1. *Shared Travel Time:* Time a vehicle transports more than one traveler,
    2. *Single Travel Time:* Time a vehicle transports exactly one traveler,
    3. *Unoccupied Travel Time:* Time a vehicle drives without a traveler, that is, empty trips,
    4. *Service Time:* Time required for travelers for getting on or off a vehicle,
    5. *Waiting Time:* Time a vehicle waits at a location for a traveler or the next request assignment.

The first metric examined is the average travel time per fulfilled request in minutes, plotted in Figure 5 against the varying fleet size. *Basic Control* creates constantly high average travel times per request even with increasing fleet size. Again, *Advanced Control* represents the counterpart, with travel time savings of 3–10 min on average, highlighting the potential of a combined advanced demand and fulfillment control for ride-sharing systems. *Advanced Demand Control* works almost as well as *Advanced Control*; only for the largest fleet size, *Advanced Fulfillment Control* becomes more efficient. Hence, the reduction of the average travel time per fulfillment is mainly rooted in demand control. Furthermore, a positive correlation can be observed between lower average travel times and the previously identified high acceptance rates, so that the reduction does not seem to be related to an overly restrictive demand control.

The second metric of interest is the pooling rate shown in Figure 5. *Basic Control* and *Advanced Control* define lower and upper bounds with a gap of 60%. Here, fulfillment control is the key for a good pooling rate as shown by the results of *Advanced*

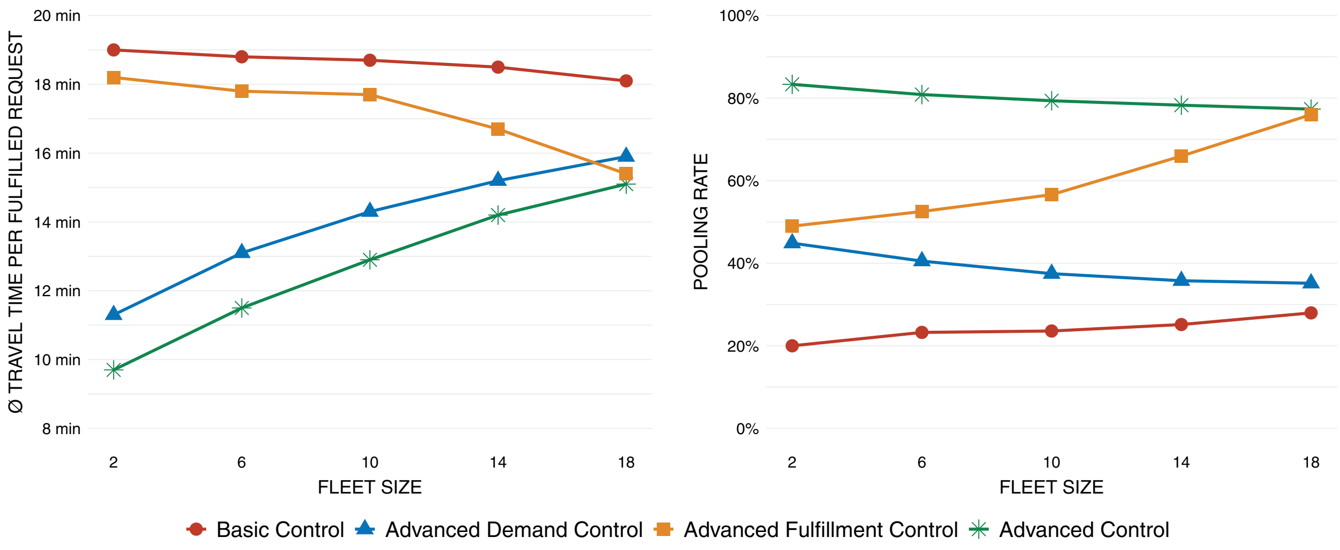

FIGURE 5 *Resource Demand Ratio*: Average travel time per fulfilled request and pooling rate

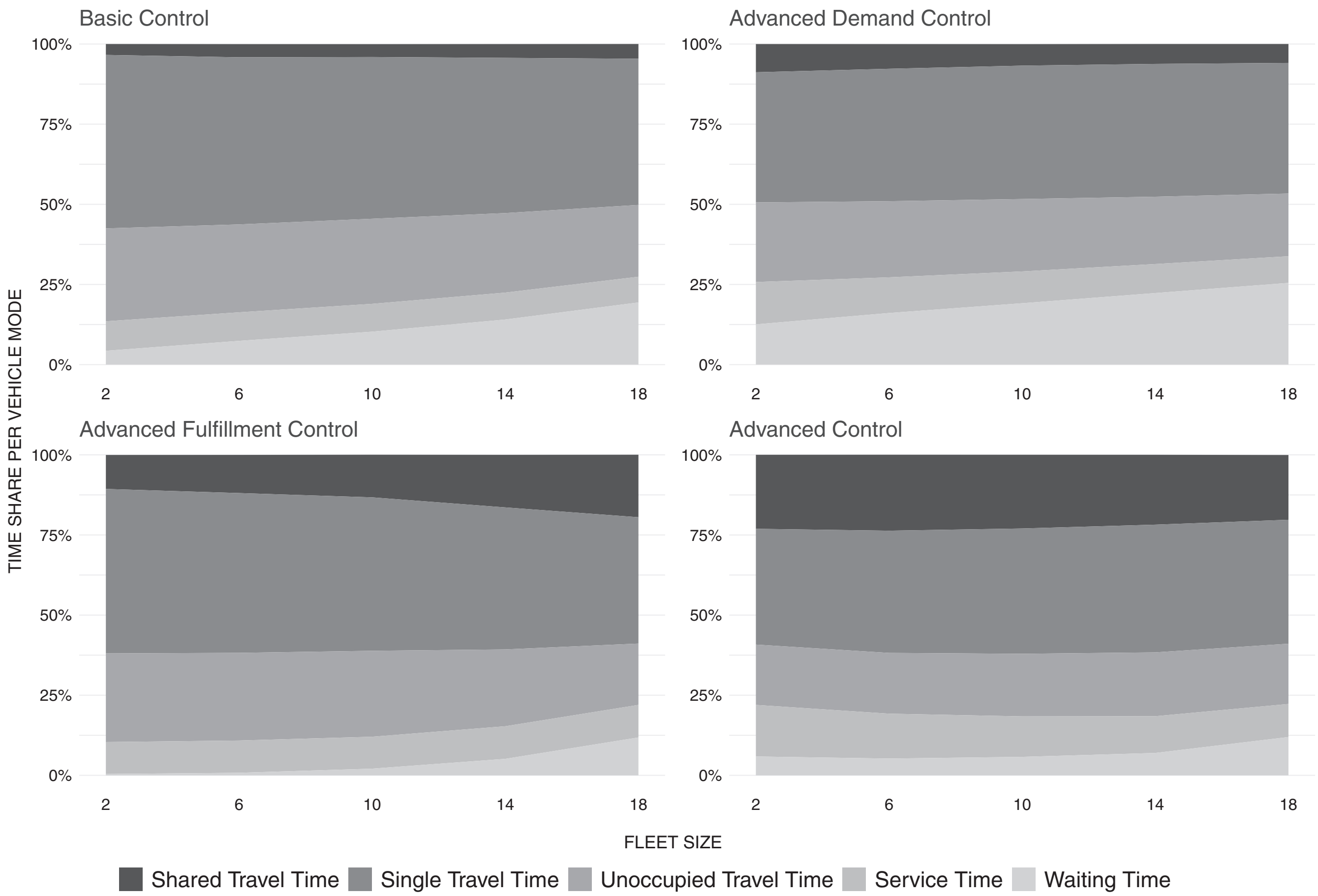

FIGURE 6 *Resource Demand Ratio*: Average time share per vehicle mode

Fulfillment Control; with increasing fleet size, it almost becomes as effective as *Advanced Control*. However, if the fleet size is small, there is a similarly high potential for improving the pooling rate through demand control.

So far, we have seen that the effectiveness of demand and fulfillment control can vary quite a bit. Advanced demand control tends to achieve a reduced average travel time per fulfillment by accepting a set of favorable requests, while advanced fulfillment control tends to offer higher pooling rates through more successful bundling of travelers. Finally, we examine the proportion of all modes a vehicle can have for the four policies (see Figure 6).

For all policies and fleet sizes, a rather stable proportion of *Unoccupied Travel Time*, as well as the relatively large share of *single travel time*, is clearly visible. Interesting differences can be observed with respect to the *Shared Travel Time* and

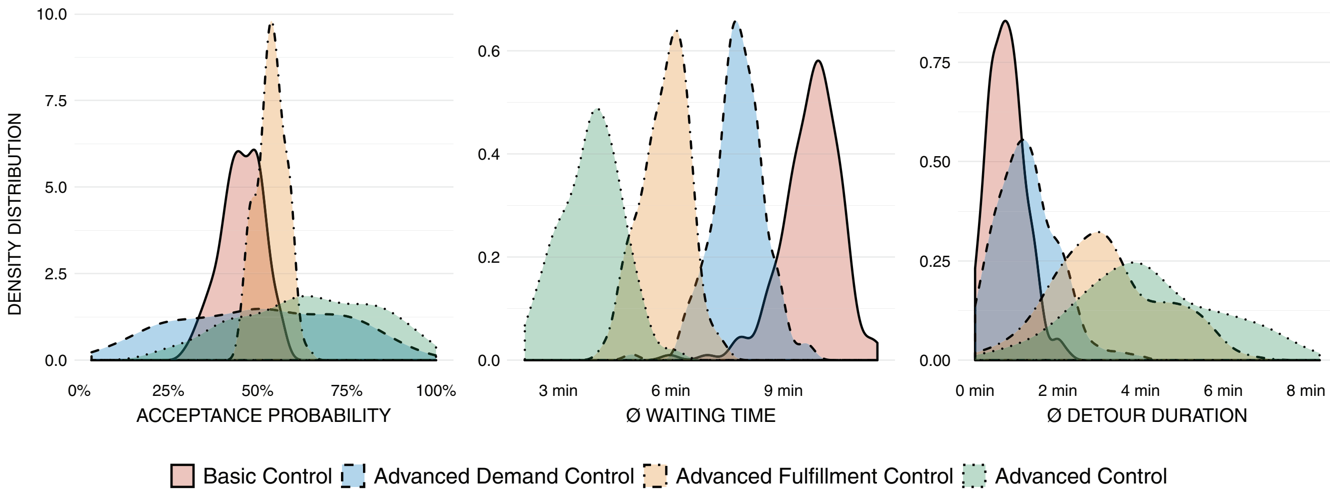

FIGURE 7 Quality of service per trip

Waiting Time. For *Shared Travel Time*, again, advanced fulfillment control seems to be the key. Interestingly, even at *Advanced Control*, only about 25% of the total fleet time is used for the simultaneous transport of more than one traveler. However, this is a significantly increased proportion compared to *Basic Control*. Major differences are also apparent for the *Waiting Time*. Especially for *Advanced Fulfillment Control* and *Advanced Control*, lower waiting times can be observed. The lower waiting times in case of an advanced fulfillment control may root in a proactive approach toward future requests. In contrast, the share of the waiting times is highest for *Advanced Demand Control*. Here, the higher waiting times arise as an advanced demand control may have vehicles wait for favorable request instead of accepting unfavorable ones. Overall, these results show different strategies regarding the handling of waiting time, whereby a smart combination of both strategies appears to be most promising

5.4 | Analysis of service quality

Finally, we examine the impact of demand and fulfillment control on the quality of service experienced by travelers. Service quality metrics are derived for each of the trips and summarized per policy. The first step is to investigate whether different service quality levels can be observed and if the trip-specific quality of service varies. We analyze the following metrics:

- The acceptance probability per trip, defined by the number of times the trip is requested divided by the number of times the request is accepted.
- The average waiting time per trip, based on the difference between the time of the request and the time the corresponding traveler is picked-up.
- The average detour duration per trip, defined as the average difference between the direct travel time of the trip and the actual time between executed pick-up and drop-off.

The results are shown in Figure 7 by means of density plots. With regard to acceptance probability, there are clear differences in the distributions. For *Basic Control* and *Advanced Fulfillment Control*, the diversification is relatively low, with a high density at about 50%. Distributions for *Advanced Demand Control* and *Advanced Control* are very flat. This indicates that the probability of being accepted is quite dissimilar among the trips regardless of the circumstances of their request, indicating that the acceptance probability depends on trip inherent characteristics. Interestingly, these characteristics seem to have a relatively minor influence on whether it is feasible to accept a trip. As seen for the analysis of acceptance probability, the average detour duration per trip also follows different distributions. What is particularly surprising is the shape of the distributions, which shows, especially for *Advanced Fulfillment Control* and *Advanced Control*, that the average detour duration varies depending on the trip. The opposite order of the distribution peaks, compared to those of the average waiting time, results from the limitation through the maximum delay parameter. The shorter waiting times achieved by an advanced demand and fulfillment control are thus partly offset by longer detours.

In the following, trip characteristics are further investigated to find correlations between acceptance probability and detour duration. To this end, we consider the location of the origin and destination as well as the distance between them. For a DVRP, Soeffker et al. [43] have already shown that anticipatory acceptance discriminates the peripheral regions of the operating area, that is, the locations there have a lower probability of acceptance. For *Advanced Demand Control*, Figure 8 illustrates this correlation separately for origin and destination of all trips, using a color scale as well as sizes that reflect the acceptance probability. The small, dark dots indicate trips with a very low acceptance probability and large, bright ones with a very high

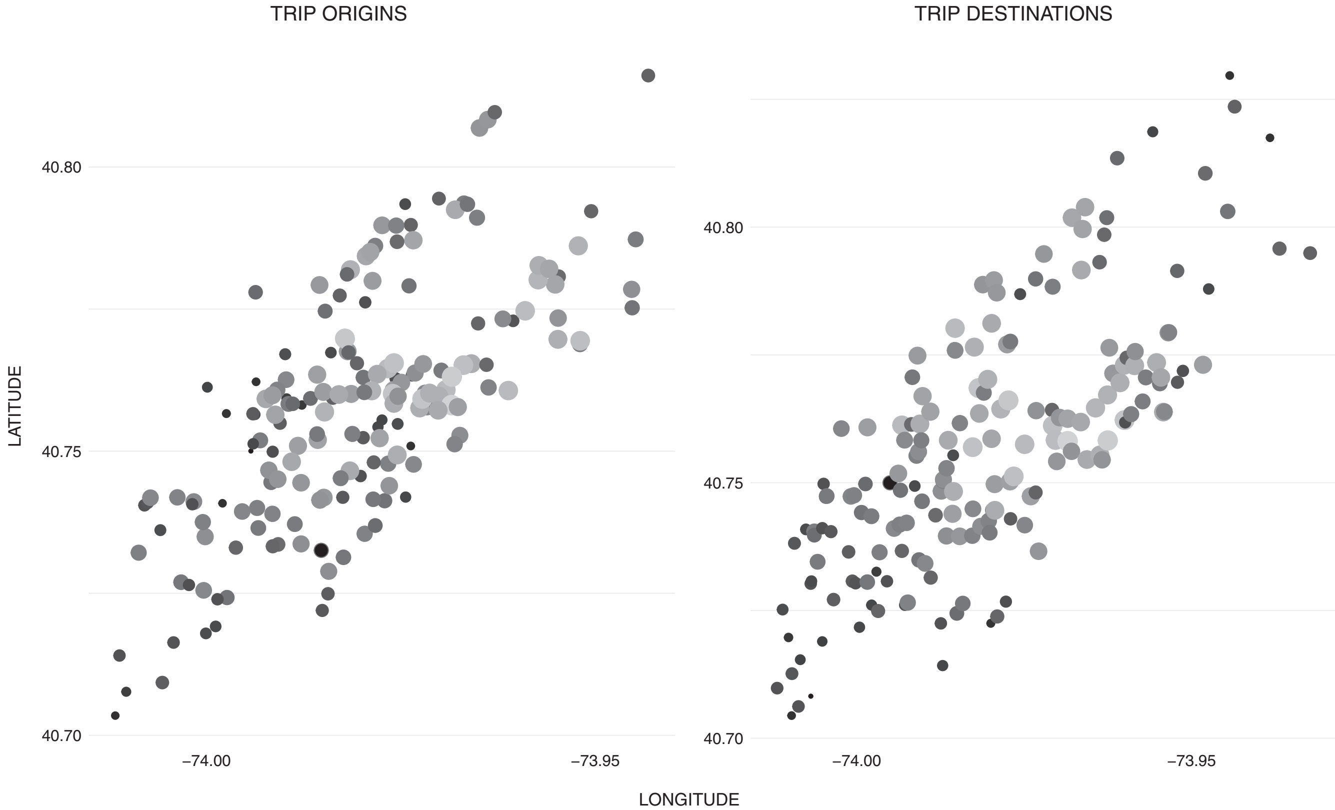

FIGURE 8 Acceptance probability per trip depending on origin and destination

acceptance probability. A preference for the regional center and the discrimination of upper and lower periphery is evident, illustrating, for demand control via enhanced acceptance decisions, the positive correlation between the acceptance probability of a trip and the geographical centrality of its origin and destination. In contrast, the analyses of average detour duration for *Advanced Demand Control* and *Advanced Fulfillment Control* did not reveal any recognizable discrimination patterns.

As a further characteristic, we examine the trip distance in the light of acceptance probability and detour duration. Results are shown in Figure 9. It becomes evident that there is a distinct negative correlation in the case of *Advanced Demand Control*. Implicitly, the advanced demand control utilizes the trip distance as a further criterion to assess requests. For the average detour per trip, a positive correlation with trip distance is noticeable for both cases. This correlation, however, is much more pronounced for *Advanced Fulfillment Control*. Hence, advanced fulfillment control penalizes long-distance trips, yet in a way that limits the usability of the ride-sharing system for such trip requests not as strict as an advanced demand control does.

In summary, demand and fulfillment control have a very different impact on the service quality of ride-sharing systems as experienced by travelers. For advanced demand control, the quality depends significantly on the nature of the requested trip. A ride-sharing system applying such a policy would be very suited for short trips in the center of the service area. However, as their requests would be rejected frequently, travelers requesting trips with unfavorable characteristics are likely to switch to other mobility services. In contrast, for advanced fulfillment control, the service would be much more balanced in terms of the acceptance probability. Yet, the increasing average detour in proportion to the distance traveled could diminish the perceived quality of service, even if this would be perceived as fair by the traveler. Finally, it should be noted that a policy exploiting the optimization potential in demand and fulfillment control would not only incorporate the performance benefits as shown in the previous sections but also the varying quality of service depending on the characteristics of the trip.

5.5 | Analysis of incomplete information

In the following, based on the *Resource Demand Ratio* sensitivity analysis, we investigate to which extent the above results change when information becomes incomplete. For this purpose, the average acceptance rates obtained under complete information are compared to those obtained for a perfect information horizon limited to the next 10 min. This analysis provides insights into the value of information defined by Mitrović-Minić et al. [28] as the "performance gap between solving an instance with incomplete and complete information."

Limiting the information horizon requires some minor adjustments to the three policies considering advanced demand and/or fulfillment control. With respect to *Advanced Demand Control*, the set of requests considered in the TOP-based favorability

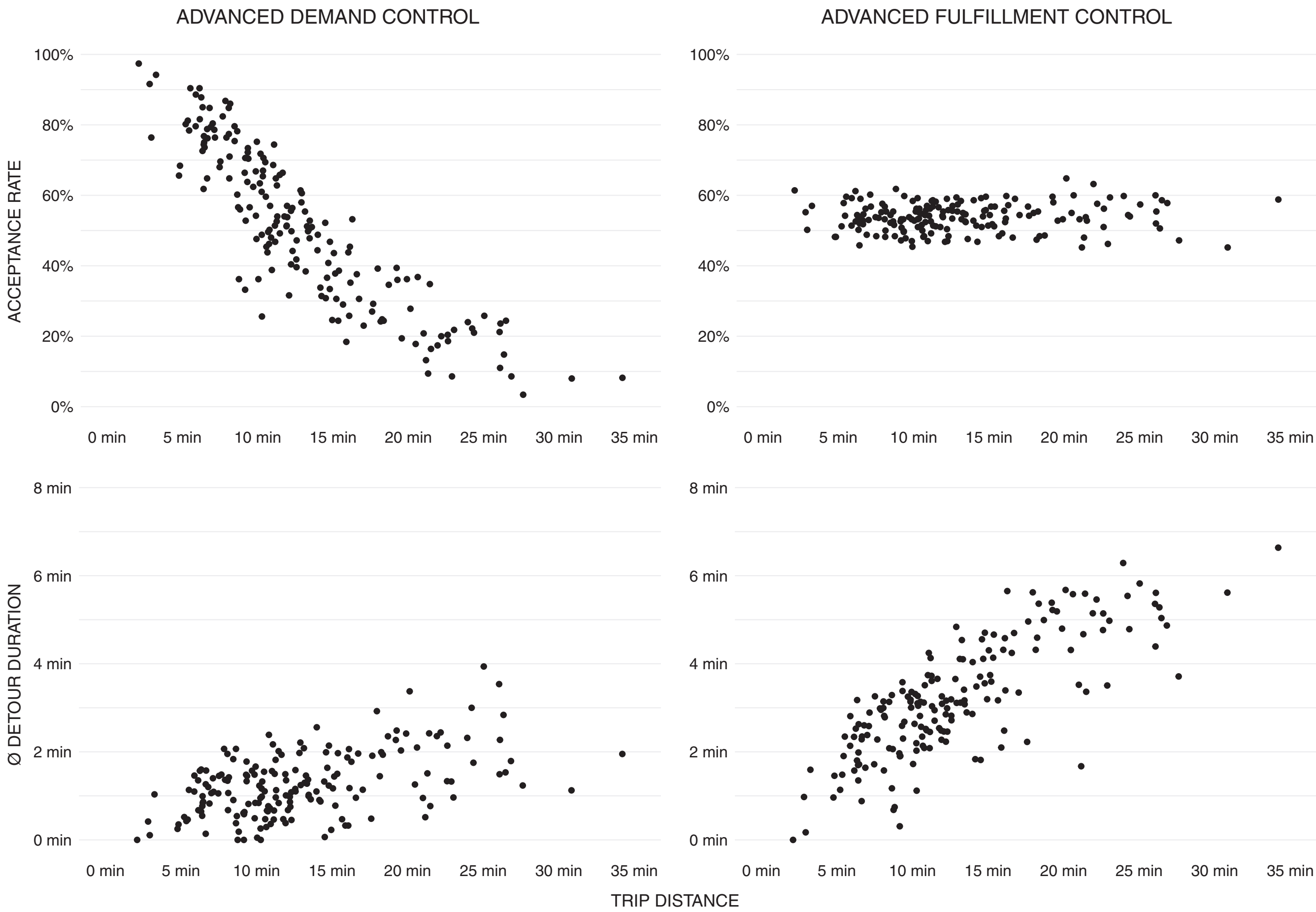

FIGURE 9 Acceptance probability and detour duration depending on the trip distances

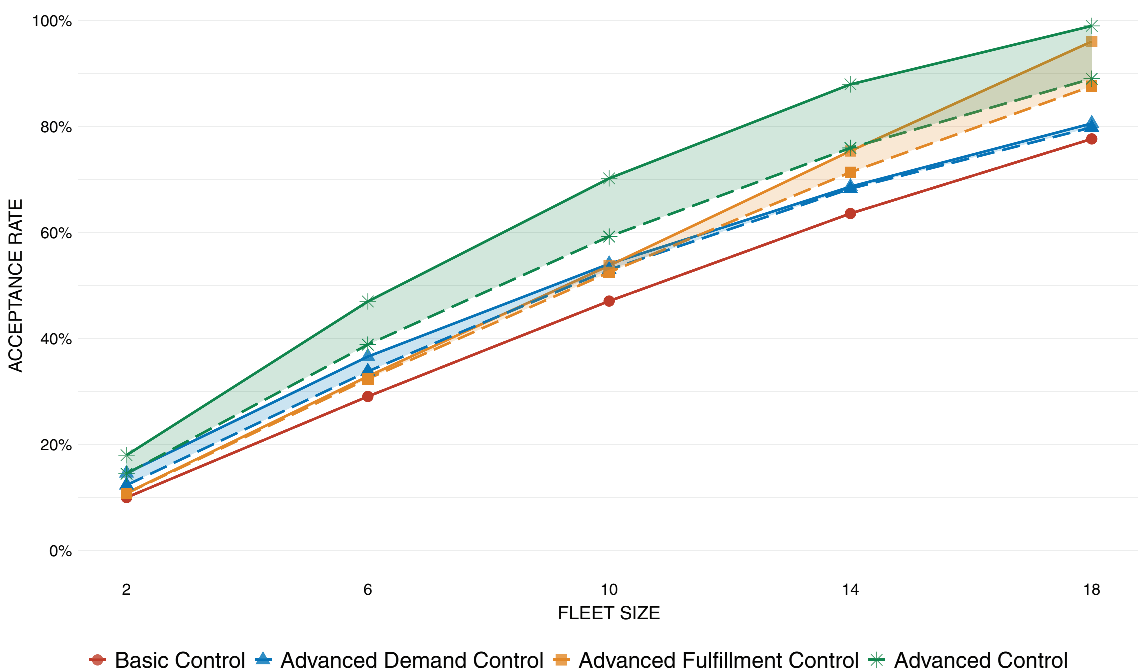

FIGURE 10 Average acceptance rates for unlimited and 10 minute information horizon

check is reduced from all future requests to those that will be received in the next 10 min. *Advanced Fulfillment Control* is adapted so that for each incoming request, feasibility checks are performed to determine which requests will be accepted in the next 10 min, to be able to re-optimize the route plan accordingly. In case of *Advanced Control*, a TOP is solved for each incoming request, taking into account the already accepted requests as well as all requests that will be received in the next 10 min to decide upon the acceptance as well as the new incumbent route plan. The results are visualized in Figure 10, which shows the acceptance rates for the four considered policies as a solid line for the unlimited information horizon and as dashed line for the information horizon limited to 10 min. The gap between the two lines of each policy is further highlighted by the respective color.

For *Basic Control*, per definition, no difference is visible, since it does not take into account any information about future demand. As expected, decreased acceptance rates can be observed in case of the limited information horizon among the other three policies. Consequently, policies that are most affected by information incompleteness are those that exploit the information most extensively. However, the structural differences between the policies implementing advanced demand and/or fulfillment control remain similar to those obtained under complete information.

These results indicate, on the one hand, that the value of information in demand and fulfillment control is directly reflected by the acceptance rates and its proportional deterioration. On the other hand, it can be observed that the previously presented findings are less pronounced in the considered case of incomplete information, yet structurally still valid. However, this investigation represents only one of various possible variations from complete over incomplete to erroneous information. It would therefore be interesting to investigate in future work under which horizon and/or quality of information the structural consistency persists and when and how it may alter.

6 | CONCLUSION

In our paper, we investigated the effectiveness of demand and fulfillment control in dynamic fleet management of ride-sharing systems. To this end, we first differentiated strategies for demand and fulfillment control and classified the related literature accordingly. Second, we defined four policies, which differ in the complexity of optimization and the amount of information exploited by demand and/or fulfillment control. The impact of these policies on dynamic fleet management was investigated in a comprehensive computational study, highlighting the operator's perspective as well as the consequences for travelers. Overall, our results demonstrated great potential for combined advanced demand and fulfillment control in dynamic fleet management. Potential benefits range from increased acceptance and pooling rates to decreased travel and idle times. However, acceptance probability and detour duration depend considerably on the nature of the requested trip.

A particular contribution of our paper is the differentiation of dynamic fleet management according to the effectiveness of demand and fulfillment control. This created insights about whether optimization potential can be attributed to either demand or fulfillment control or a reasonable combination of them. This is important since advanced demand and fulfillment control differ in their requirements as well as in the effect on the performance of the ride-sharing system. Advanced demand control is especially beneficial if there is a sufficient surplus of demand, that is, when there is a decent subset of favorable requests that can be selected from a larger pool of feasible requests. Furthermore, advanced demand control can increase the acceptance rate primarily through a significant decrease of average travel time per fulfilled request. The acceptance probability is highly correlated with the nature of the requested trip, leading to an acceptance of short trips that are centrally located in the service area. The potential of fulfillment control is primarily associated with the acceptance rate and the promised fulfillment time window. Taking acceptance of future requests into account, advanced fulfillment control enables a rather stable performance despite increasingly narrow fulfillment time windows. However, advanced fulfillment control can only be beneficial if demand control has only a minor impact or is of advanced nature, too. In particular, performance improvement through advanced fulfillment control can be traced back to a much more successful bundling of requests. The consequence for travelers is that the detour duration increases proportionally to the distance of the trip.

Our paper offers operators of ride-sharing systems an orientation on how to implement demand and fulfillment control. For instance, advanced demand control could be more suitable for large systems or systems with a few regular travelers, where the satisfaction of individual travelers is negligible. Furthermore, it could be implemented in order to efficiently manage a temporary demand surplus on special occasions. Advanced fulfillment control would be particularly suitable for systems with stable demand allowing precise anticipation of future acceptances. Moreover, under the assumption that a selective demand control avoids, the demand target should be fully operable.

Moreover, we contribute to research on dynamic fleet management by providing a more differentiated view of how policies control demand and fulfillment in a ride-sharing system. We believe that this can be the basis for a better understanding of the varying effectiveness of existing policies as well as the development of new ones.

In the future, a more detailed overview of anticipatory decision-making in dynamic fleet management could provide a better understanding of what types of anticipation are reasonable for ride-sharing systems. Furthermore, for our study, we performed the evaluation mostly assuming complete information while the implications of incomplete information were only briefly examined. An intuitive next step would be to preform demand and fulfillment control under different degrees of incomplete or imperfect information to investigate the link between information quality and the exploitation of the identified potentials. Moreover, state-of-the-art policies for demand and/or fulfillment control could be evaluated to compare their effectiveness with the results obtained. This would include the development of sophisticated customer acceptance mechanisms for anticipatory demand control in ride-sharing systems.

ACKNOWLEDGEMENTS

This research was supported by a grant from the German Federal Ministry of Transport and Digital Infrastructure (BMVI, Grant No. 16AVF2147E).
Open access funding enabled and organized by Projekt DEAL.

DATA AVAILABILITY STATEMENT

Data available on request from the authors.

ORCID

Jarmo Haferkamp https://orcid.org/0000-0002-5914-7539
Jan Fabian Ehmke https://orcid.org/0000-0001-8474-7483

REFERENCES

[1] J. Alonso-Mora, S. Samaranayake, A. Wallar, E. Frazzoli, and D. Rus, *On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment*, Proc. Natl. Acad. Sci **114**(3) (2017a), 462–467.
[2] J. Alonso-Mora, A. Wallar, and D. Rus, *Predictive routing for autonomous mobility-on-demand systems with ride-sharing*, Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE; 2017b, pp. 3583–3590.
[3] A. Attanasio, J. F. Cordeau, G. Ghiani, and G. Laporte, *Parallel tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem*, Parallel Comput. **30**(3) (2004), 377–387.
[4] A. Beaudry, G. Laporte, T. Melo, and S. Nickel, *Dynamic transportation of patients in hospitals*, OR Spectr. **32**(1) (2010), 77–107.
[5] R. W. Bent and P. van Hentenryck, *Scenario-based planning for partially dynamic vehicle routing with stochastic customers*, Oper. Res. **52**(6) (2004), 977–987.
[6] G. Berbeglia, G. Pesant, and L. M. Rousseau, *Checking the feasibility of dial-a-ride instances using constraint programming*, Transp. Sci. **45**(3) (2011), 399–412.
[7] G. Berbeglia, J. F. Cordeau, and G. Laporte, *A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem*, INFORMS J. Comput. **24**(3) (2012), 343–355.
[8] J. Branke, M. Middendorf, G. Noeth, and M. Dessouky, *Waiting strategies for dynamic vehicle routing*, Transp. Sci. **39**(3) (2005), 298–312.
[9] A. M. Campbell and M. W. P. Savelsbergh, *Decision support for consumer direct grocery initiatives*, Transp. Sci. **39**(3) (2005), 313–327.
[10] A. M. Campbell and M. Savelsbergh, *Incentive schemes for attended home delivery services*, Transp. Sci. **40**(3) (2006), 327–341.
[11] I. M. Chao, B. L. Golden, and E. A. Wasil, *The team orienteering problem*, Eur. J. Oper. Res. **88**(3) (1996), 464–474.
[12] L. Coslovich, R. Pesenti, and W. Ukovich, *A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem*, Eur. J. Oper. Res. **175**(3) (2006), 1605–1615.
[13] G. Cwioro, P. Hungerländer, K. Maier, J. Pöcher, and C. Truden, *An optimization approach to the ordering phase of an attended home delivery service*, in *Integration of constraint programming, artificial intelligence, and operations research theoretical computer science and general issues*, L. M. Rousseau and K. Stergiou, Eds., Springer International Publishing, Cham, Switzerland, 2019, 208–224.
[14] R. B. Dial, *Autonomous dial-a-ride transit introductory overview*, Transp. Res. C Emerg. Technol. **3**(5) (1995), 261–275.
[15] J. F. Ehmke and A. M. Campbell, *Customer acceptance mechanisms for home deliveries in metropolitan areas*, Eur. J. Oper. Res. **233**(1) (2014), 193–207.
[16] GraphHopper, *GraphHopper routing engine*. n.d., available at https://github.com/graphhopper/graphhopper.
[17] S. C. Ho, W. Y. Szeto, Y. H. Kuo, J. M. Y. Leung, M. Petering, and T. W. H. Tou, *A survey of dial-a-ride problems: Literature review and recent developments*, Transp. Res. B Methodol. **111** (2018), 395–421.
[18] M. E. T. Horn, *Fleet scheduling and dispatching for demand-responsive passenger services*, Transp. Res. C Emerg Technol **10**(1) (2002), 35–63.
[19] H. Hosni, J. Naoum-Sawaya, and H. Artail, *The shared-taxi problem: Formulation and solution methods*, Transp. Res. B Methodol. **70** (2014), 303–318.
[20] E. Hyytiä, A. Penttinen, and R. Sulonen, *Non-myopic vehicle and route selection in dynamic DARP with travel time and workload objectives*, Comput. Oper. Res. **39**(12) (2012), 3021–3030.
[21] S. Ichoua, M. Gendreau, and J. Y. Potvin, *Exploiting knowledge about future demands for real-time vehicle dispatching*, Transp. Sci. **40**(2) (2006), 211–225.
[22] M. Lowalekar and P. Jaillet, *ZAC: A zone path construction approach for effective real-time ridesharing*, Proceedings of the International Conference on Automated Planning and Scheduling; Vol. 29, 2019, pp. 528–538.
[23] K. Lund, O. Madsen, and J. M. Rygaard, *Vehicle routing problems with varying degrees of dynamism. Technical report*, Institute of Mathematical Modelling.
[24] S. Ma, Y. Zheng, and O. Wolfson, *T-share: A large-scale dynamic taxi ridesharing service*, Proceedings of 2013 IEEE 29th International Conference on Data Engineering (ICDE), April 8–April 12, 2013, pp. 410–421.

[25] J. Mackert, *Choice-based dynamic time slot management in attended home delivery*, Comput. Ind. Eng. **129** (2019), 333–345.
[26] O. B. G. Madsen, H. F. Ravn, and J. M. Rygaard, *A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives*, Ann. Oper. Res. **60**(1) (1995), 193–208.
[27] S. Mitrović-Minić and G. Laporte, *Waiting strategies for the dynamic pickup and delivery problem with time windows*, Transp. Res. B Methodol. **38**(7) (2004), 635–655.
[28] S. Mitrović-Minić, R. Krishnamurti, and G. Laporte, *Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows*, Transp. Res. B Methodol. **38**(8) (2004), 669–685.
[29] Y. Molenbruch, K. Braekers, and A. Caris, *Typology and literature review for dial-a-ride problems*, Ann. Oper. Res. **259**(1–2) (2017), 295–325.
[30] NYC Taxi and Limousine Commission, *Trip record data*; available at https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.
[31] D. Pisinger and S. Ropke, *Large neighborhood search*, in *Handbook of metaheuristics, volume. 146 of International Series in Operations Research & Management Science*, M. Gendreau and J. Y. Potvin, Eds., Springer, Boston, MA, 2010, 399–419.
[32] J. Y. Potvin and J. M. Rousseau, *A parallel route building algorithm for the vehicle routing and scheduling problem with time windows*, Eur. J. Oper. Res. **66**(3) (1993), 331–340.
[33] M. Pouls, A. Meyer, and N. Ahuja, *Idle vehicle repositioning for dynamic ride-sharing*, in *Computational logistics theoretical computer science and general issues*, E. Lalla-Ruiz, M. Mes, and S. Voß, Eds., Springer International Publishing and Imprint: Springer, Cham, Switzerland, 2020, 507–521.
[34] H. N. Psaraftis, M. Wen, and C. A. Kontovas, *Dynamic vehicle routing problems: Three decades and counting*, Networks **67**(1) (2016), 3–31.
[35] H. Qiu, R. Li, and J. Zhao, *Dynamic pricing in shared mobility on demand service*. available at https://arxiv.org/pdf/1802.03559.
[36] C. Riley, A. Legrain, and P. van Hentenryck, *Column generation for real-time ride-sharing operations*, in *Integration of constraint programming, artificial intelligence, and operations research*, Lecture Notes in Computer Science, Vol. **11494**, L. M. Rousseau and K. Stergiou, Eds., Springer International Publishing, Cham, Switzerland, 2019, 472–487.
[37] C. Riley, P. van Hentenryck, and E. Yuan, *Real-time dispatching of large-scale ride-sharing systems: Integrating optimization, machine learning, and model predictive control*, Proceedings of the 29th International Joint Conference on Artificial Intelligence; Vol. 5, 2020, pp. 4417–4423.
[38] U. Ritzinger, J. Puchinger, and R. F. Hartl, *A survey on dynamic and stochastic vehicle routing problems*, Int. J. Prod. Res. **54**(1) (2016), 215–231.
[39] S. Ropke and D. Pisinger, *An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows*, Transp. Sci. **40**(4) (2006), 455–472.
[40] M. Schilde, K. F. Doerner, and R. F. Hartl, *Metaheuristics for the dynamic stochastic dial-a-ride problem with expected return transports*, Comput. Oper. Res. **38**(12) (2011), 1719–1730.
[41] S. Shah, M. Lowalekar, and P. Varakantham, *Neural approximate dynamic programming for on-demand ride-pooling*, Proc AAAI Conf Artif Intell **34**(01) (2020), 507–515.
[42] P. Shaw, *Using constraint programming and local search methods to solve vehicle routing problems*, in *Principles and practice of constraint programming — CP98. volume. 1520 of Lecture Notes in Computer Science*, G. Goos, J. Hartmanis, J. van Leeuwen, M. Maher, and J. F. Puget, Eds., Springer, Berlin/Heidelberg, Germany, 1998, 417–431.
[43] N. Soeffker, M. W. Ulmer, and D. C. Mattfeld, *On fairness aspects of customer acceptance mechanisms in dynamic vehicle routing*, in *Proceedings of logistikmanagement*, R. O. Large, N. Kramer, A.-K. Radig, M. Schäfer, and A. Sulzbach, Eds., 2017, 17–24.
[44] B. W. Thomas, *Waiting strategies for anticipating service requests from known customer locations*, Transp. Sci. **41**(3) (2007), 319–331.
[45] Uber Pool, *How uber pool works*. n.d., available at https://www.uber.com/us/en/ride/uberpool/.
[46] M. W. Ulmer, D. C. Mattfeld, and F. Köster, *Budgeting time for dynamic vehicle routing with stochastic customer requests*, Transp. Sci. **52**(1) (2018), 20–37.
[47] M. W. Ulmer, J. C. Goodson, D. C. Mattfeld, and M. Hennig, *Offline–online approximate dynamic programming for dynamic vehicle routing with stochastic requests*, Transp. Sci. **53**(1) (2019), 185–202.
[48] Z. Xiang, C. Chu, and H. Chen, *The study of a dynamic dial-a-ride problem under time-dependent and stochastic environments*, Eur. J. Oper. Res. **185**(2) (2008), 534–551.
[49] X. Yang and A. K. Strauss, *An approximate dynamic programming approach to attended home delivery management*, Eur. J. Oper. Res. **263**(3) (2017), 935–945.
[50] X. Yang, A. K. Strauss, C. S. M. Currie, and R. Eglese, *Choice-based demand management and vehicle routing in e-fulfillment*, Transp. Sci. **50**(2) (2016), 473–488.
[51] X. Yu and S. Shen, *An integrated decomposition and approximate dynamic programming approach for on-demand ride pooling*, IEEE Trans. Intell. Transp. Syst. **21**(9) (2020), 3811–3820.

How to cite this article: J. Haferkamp, and J. F. Ehmke, *Effectiveness of demand and fulfillment control in dynamic fleet management of ride-sharing systems*, Networks. **79** (2022), 314–337. https://doi.org/10.1002/net.22062

APPENDIX

A.1 | Parameter tuning

The parameter tuning of the LNS is based on the *Resource Demand Ratio* sensitivity analysis. Ten instances generated for parameter tuning are solved five times, each time with an adapted fleet size. For feasibility check and re-optimization in the scope of *Basic Control*, and *Advanced Demand Control* the tuning of the parameters is based on *Basic Control*. For *Advanced Fulfillment Control*, a separated tuning is performed, since considerably more requests have to be handled during a feasibility check and the final optimization. Regarding the TOP, the parameter tuning is based on *Advanced Control*. The resulting values are mostly applied as well to solve the TOP as favorability check within *Advanced Demand Control*. However, the number of required iterations β and thus the computational effort is determined separately.

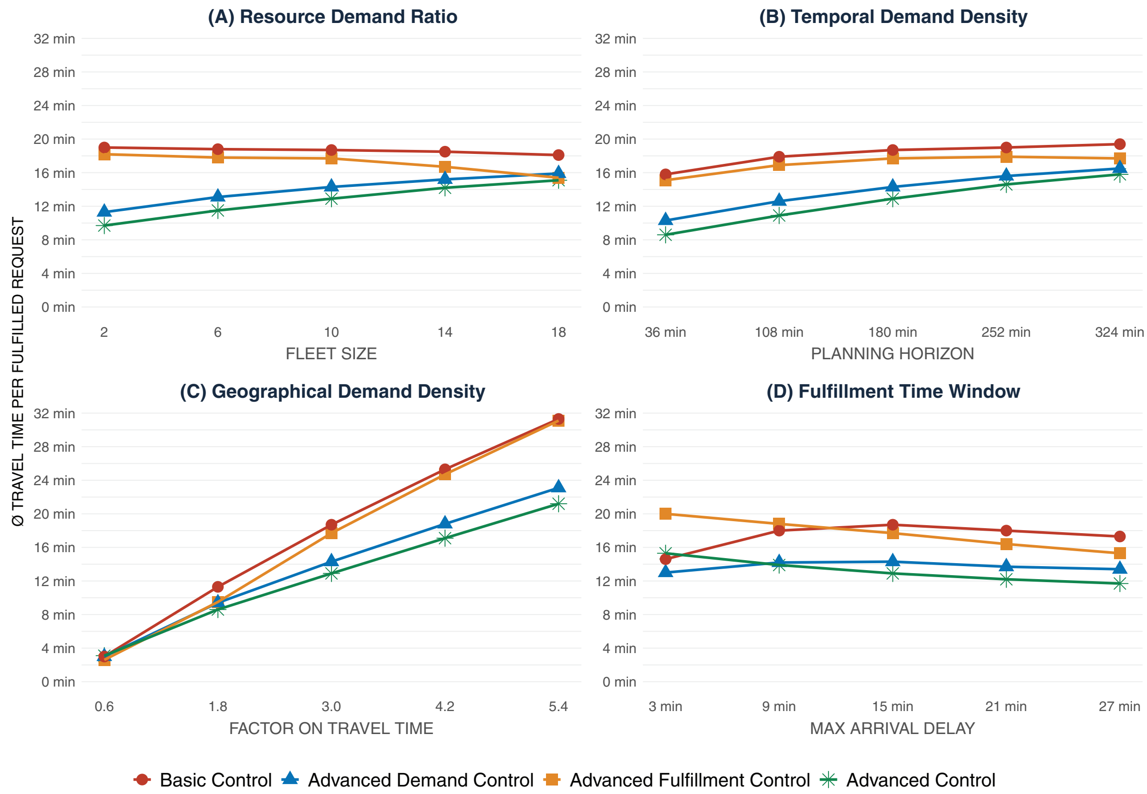

FIGURE A1 Sensitivity analyses: Average travel time per fulfilled request

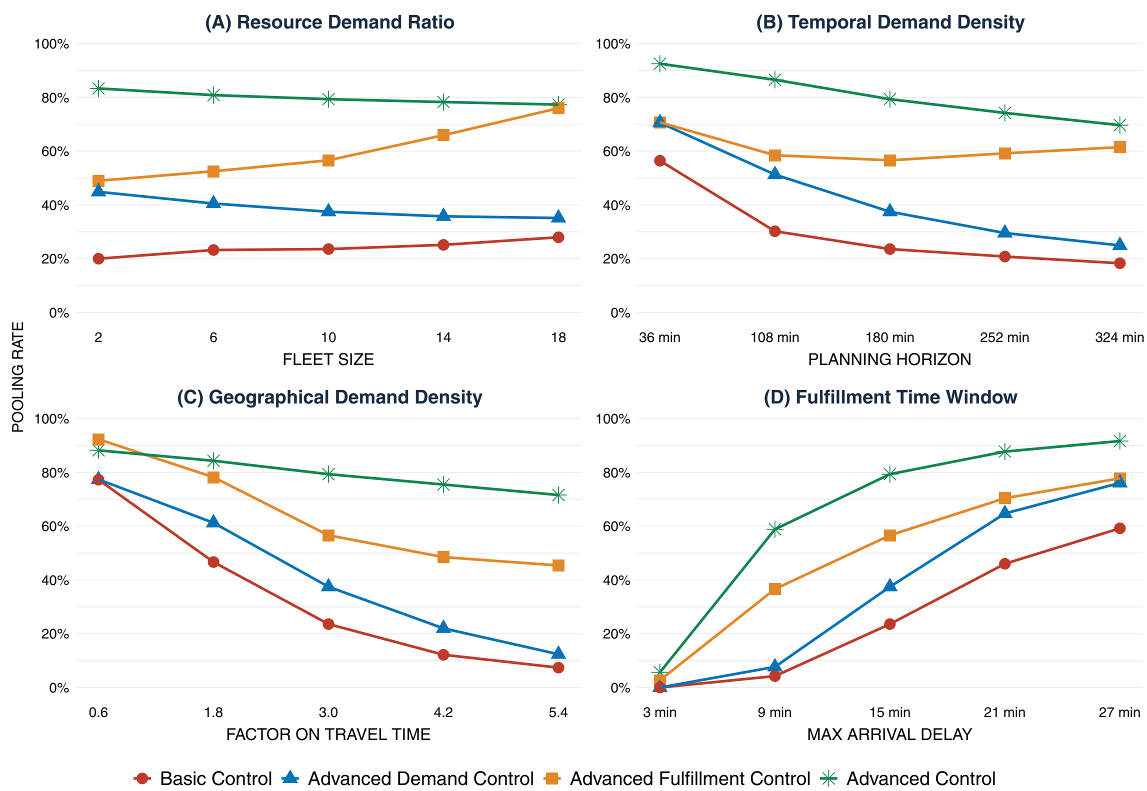

FIGURE A2 Sensitivity analyses: Pooling rate

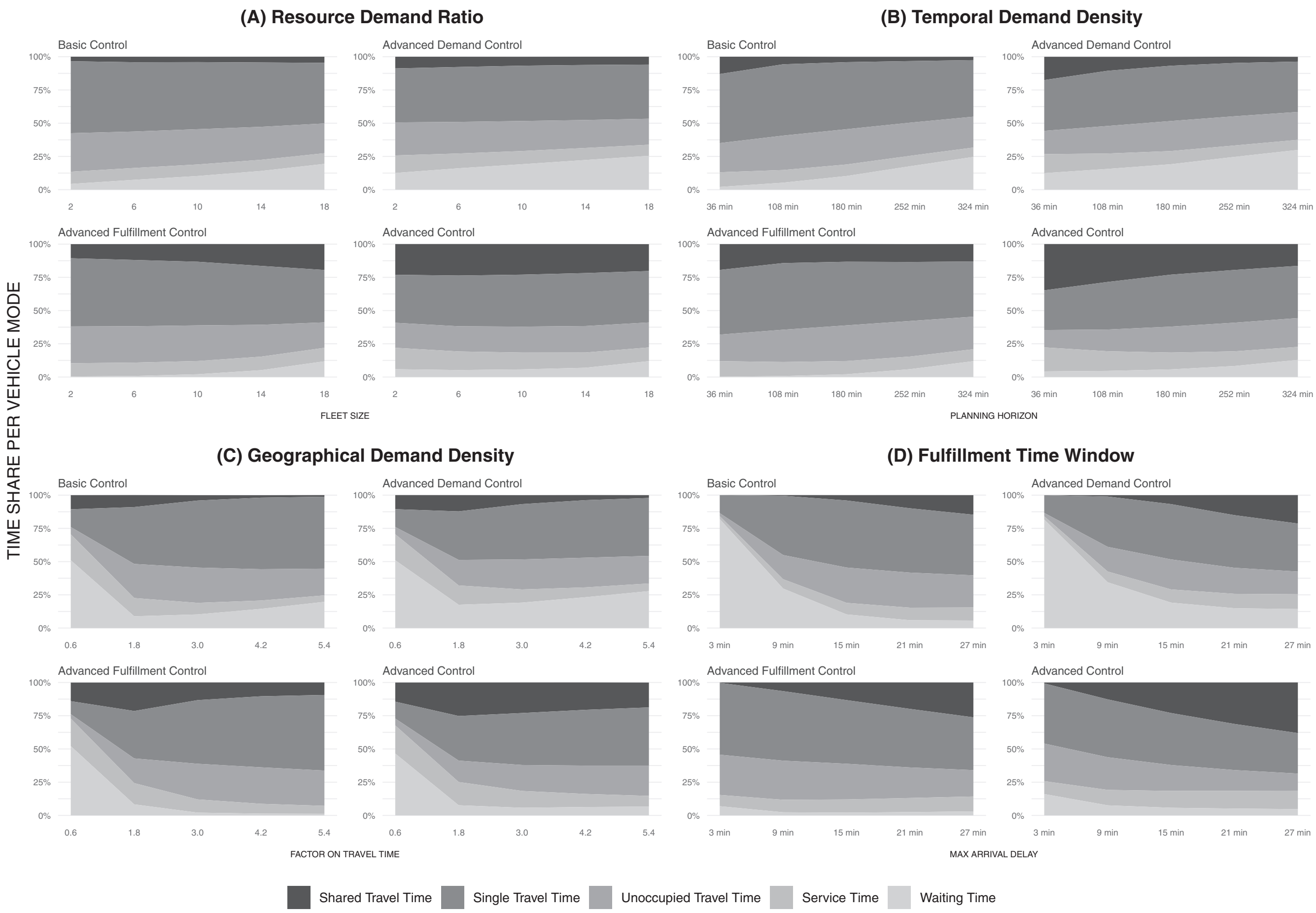

FIGURE A3 Sensitivity analyses: Time share per vehicle mode

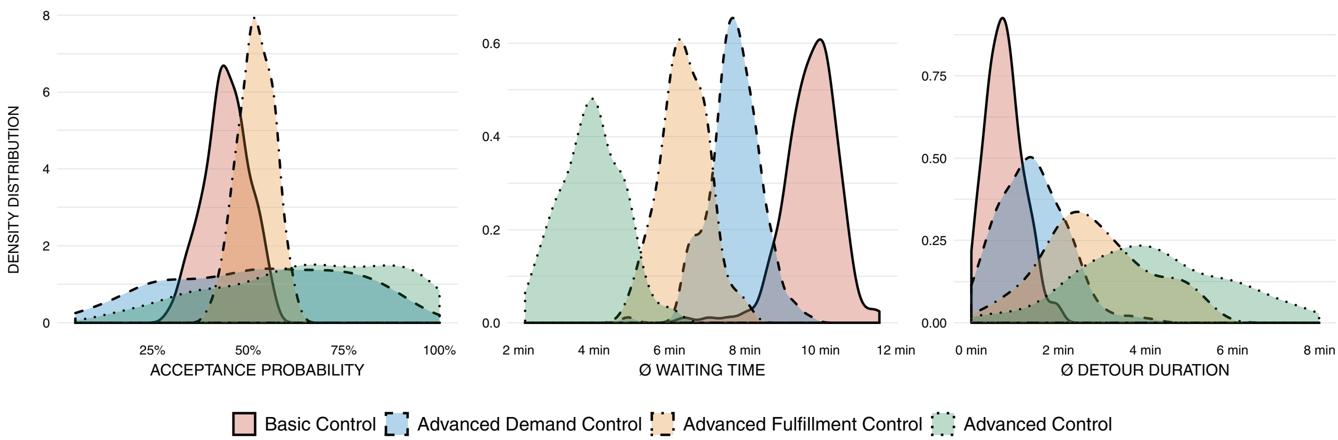

FIGURE A4 *Temporal Demand Density:* Quality of service per trip

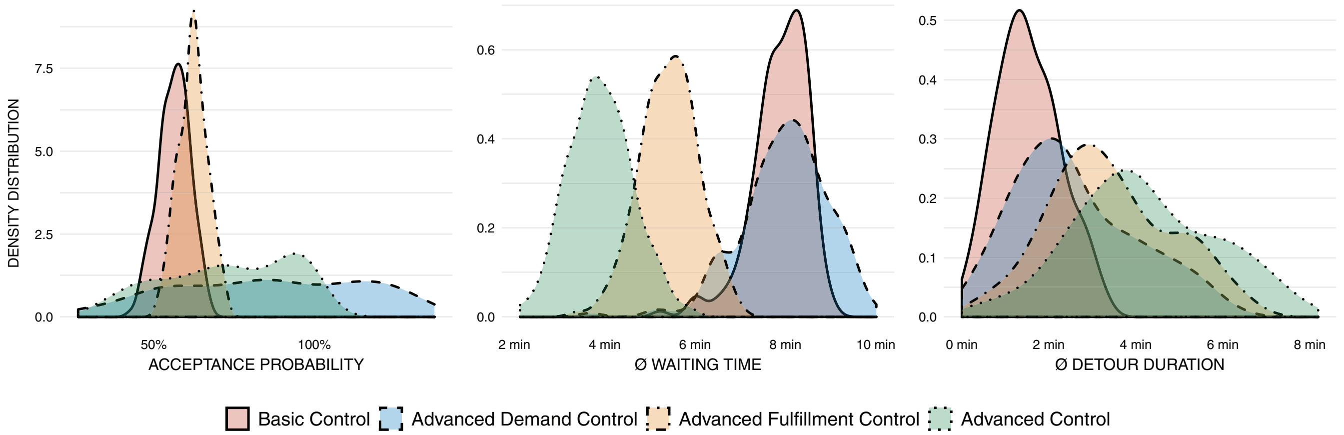

FIGURE A5 *Geographical Demand Density:* Quality of service per trip

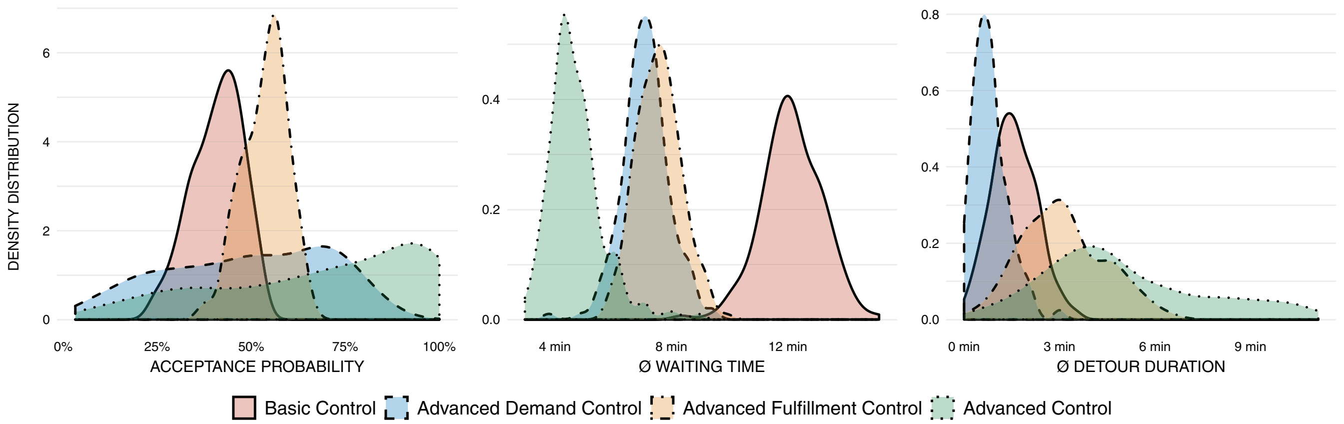

FIGURE A6 *Fulfillment Time Window:* Quality of service per trip

TABLE A1 Number of iterations

				ø Last successful iteration per fleet size				
Policy	**Case**	**Final β**	**Test β**	**2**	**6**	**10**	**14**	**18**
Basic Control	Feasibility check	100	1000	1	2	5	5	**11**
	Re-optimization	200	1000	0	1	8	24	**126**
Advanced Demand Control	TOP	3000	3000	2895	2995	2995	2996	**2999**
Advanced Fulfillment Control	Feasibility check	1000	2000	260	531	728	**910**	719
	Final optimization	10 000	10 000	3	1245	6283	9713	**9778**
Advanced Control	TOP	30 000	40 000	4770	15195	15\,288	**27 235**	15 306

Note: The values are highlighted on the basis of which one is varied in each of the analyses.
Abbreviation: TOP, team orienteering problem.

The number of iterations as termination criterion has a particular impact on the solution quality and the computing time. We define a reasonable maximum number of iterations β as follows. We begin with an overly large number and then check the last iteration yielding a new best solution. The final number of iterations is then determined in dependence of its magnitude by rounding up to the next number divisible by 100, 1000, or 10 000. The results of this procedure are summarized in Table A1. At the beginning, the percentage of trips removed per iteration is set to $\gamma_1 = 0.3$ and $\gamma_2 = 0.4$ following Ropke and Pisinger [39], and the noise for the operators is set to a medium level of $\delta_1 = 4$ and $\delta_2 = 4$.

It can be observed that the values vary considerably, which is due to the different number of replannable requests and the differences between single and repeated execution. Overall, a reasonable value of β could be determined for most of the cases. An exception is the TOP in case of *Advanced Demand Control*. Here, improvements are still found for all fleet sizes close to the last iteration. A further increase of the number of iterations was omitted, since the tested β values already induce significant computational effort. However, since this check is simply intended to determine whether a trip is favorable, that is,

TABLE A2 Percentage of requests removed per iteration

$\gamma_1 - \gamma_2$	Advanced control TOP	Advanced fulfillment control Feasibility check and final optimization	Basic control Feasibility check and re-optimization
10%–20%	**64.4%**	**54.0%**	44.4%
30%–40%	63.9%	53.3%	44.5%
50%–60%	61.8%	51.5%	44.5%
70%–80%	61.4%	50.5%	**44.6%**

Note: The values are highlighted on the basis of which one is varied in each of the analyses.
Abbreviation: TOP, team orienteering problem.

TABLE A3 Noise value regret-2 insertion

Values	Advanced control TOP	Advanced fulfillment control Feasibility check and final optimization	Basic Control Feasibility check and re-optimization
$\delta_1 = 0$	64.2%	53.8%	44.6%
$\delta_1 = 4$	**64.4%**	**54.0%**	**44.7%**
$\delta_1 = 8$	64.3%	53.6%	44.6%

Note: The values are highlighted on the basis of which one is varied in each of the analyses.

TABLE A4 Noise value worst-removal

Values	Advanced control TOP	Advanced fulfillment control Feasibility check and final optimization	Basic control Feasibility check and re-optimization
$\delta_2 = 0$	64.3%	53.7%	44.6%
$\delta_2 = 4$	**64.4%**	**54.0%**	**44.7%**
$\delta_2 = 8$	64.4%	53.8%	44.6%

Note: The values are highlighted on the basis of which one is varied in each of the analyses.
Abbreviation: TOP, team orienteering problem.

whether it can be easily integrated together with current and future requests, there is no need to focus on exceptional solution quality.

Further parameter values are determined by the acceptance rate calculated across all instances. The first parameter values are γ_1 and γ_2, which control the minimum and maximum percentage of requests to be removed per iteration. To determine these two parameters, values between $\gamma_1 = 0.1$, $\gamma_2 = 0.2$ and $\gamma_1 = 0.7$, $\gamma_2 = 0.8$ were tested for the same LNS cases as before. It turns out that in cases with a high number of replannable requests, lower values and thus smaller changes in the solution are advantageous. The acceptance rate for these cases differs up to 4%. In the opposite case, with only a few replannable requests, higher values are slightly advantageous, however, the differences are small. Based on these results, for the insertion and re-optimization in the case of *Basic Control* and *Advanced Demand Control*, $\gamma_1 = 0.7, \gamma_2 = 0.8$ is applied. For both TOP as well as the insertion and final optimization of *Advanced Fulfillment Control*, we set $\gamma_1 = 0.1, \gamma_2 = 0.2$. Regarding the noise parameters δ_1 and δ_2, which are applied in the worst-removal and the regret-2 operator, no noise (δ_1 and $\delta_2 = 0$), medium noise (δ_1 and $\delta_2 = 4$) and a high degree of noise (δ_1 and $\delta_2 = 8$) are examined separately. However, a significant influence on the acceptance rate could not be determined. Since the results were best for all examined cases when using a medium noise (δ_1 and $\delta_2 = 4$), this value is selected for the experiments. For detailed results of the tuning of γ_1, γ_2, δ_1, and δ_2 see Tables A2, A3, and A4.