The online steady-state technician booking problem

Edison Avraham, Tal Raviv
Department of Industrial Engineering, Tel Aviv University
October 2020

Abstract

We present a model for policy optimization of mobile personnel's online booking over a multiday horizon, with a different cutoff for each day. The system interacts with each customer, wishing to place a service request with the goal of maximizing the expected ratio of accepted requests in the long term. The demand forecast and user choice model are inputs of the policy optimization problem. The interactions with the customers are performed in a single step: The system offers an assortment of time slots during the service horizon, and the user either chooses one of them or abandons the system. The model maintains a tentative routing and scheduling solution that is updated after the acceptance of each request. The assortment of time slots following each service request is constructed by maximizing the expected net gain from the assortment. The net gain is estimated using a Cobb-Douglass function of features that represent the system's current state in a concise way. The parameters of this function are fitted using a simulation framework. The proposed method is benchmarked based on randomly generated datasets in various demand scenarios and geographies. The method is shown to significantly outperform a more straightforward baseline policy that is commonly used.

Keywords: Vehicle Routing, Field Service Management, Booking

1 Introduction

In recent years, the operation of mobile personnel that visits customers at their locations has attracted considerable interest. The first stage of the service provision process involves an interaction between the service provider and the customer regarding the time slot for the beginning of the service at the customer location. The provider should take into account spatial and temporal considerations when agreeing to certain time slots while the customer has her considerations that are unknown to the provider. In this work, we refer to this interaction as the booking process. We envision an online process in which the provider side is fully automated.

Specifically, as described in Section 3, we define the online steady-state booking (O-SSTBP) problem, in which the provider is the leader in the booking process. That is, the provider offers several possible time slots to the customer. The offered time slots may span over several subsequent working days. The customer decides whether to accept one of the offered slots and place a service request or to abandon the system. The objective of the provider is to find an optimal policy by which an assortment of offers is constructed online, each time a customer enters the system. The service provider's goal is to maximize the ratio of accepted requests in the long run. Note that we do not explicitly include the travel time of the mobile personnel in the objective function. However, a policy that maximizes the number of
served customers within the predefined time slots implicitly minimizes the travel time. We use the term technician to describe the mobile personnel that visit the customer and provide the service. However, the problem is relevant for many other types of services.

During the booking process, the service provider should consider the following trade-off. By offering many possible time slots, the provider can increase the customer's probability of accepting one of them but at the cost of a greater chance that the customer selects a slot that requires greater travel time. Consequently, the availability of the service provider to future customer requests is reduced.

We note that the O-SSTBP arises in many practical situations where service requests can be placed a few days in advance. The provider cannot control the demand by dynamically setting the service fees but can control the availability of the time slots for service. For example, if the service is provided under some maintenance agreement or warranty.

The service horizon is divided into time slots of 2–3 hours, for which requests are assigned. Once the provider commits to serve a customer within a particular slot, she must respect the commitment. The customers have a predefined preference over the slots, and this preference consists of two components: a common component to all the customers, which is known to the provider, and an individual random component. We use a discrete choice model to describe the customer decision. The multinomial logit (MNL) model is used for demonstration in our numerical study.

For each time slot, we estimate the opportunity cost of serving the current customer, in terms of the expected number of customers that will be lost if the current customer accepts the service offer. This estimate is obtained by fitting the parameters of a Cobb–Douglas function with the following arguments: (1) the remaining idle time in the slot of the technician after the customer request is inserted in the slot; (2) the remaining idle time in the daily route of the technician after the customer request is inserted in the slot; (3) the additional travel time required to serve the request. The advantage of using a Cobb–Douglas function to represent the opportunity cost lay in the fact that it allows capturing the interactions between the above measures. E.g., a long remaining time in the day is not enough to indicate a low opportunity cost of a slot if it is fully utilized.

The expected net gain from an offered slot accepted by a customer is one minus its opportunity cost. The assortment of slots offered by the provider is a subset of the available slots, which maximizes the expected net gain based on the customers’ discrete choice model. We use a simulation to estimate the parameters of the opportunity cost function. The assortment problem is solved whenever a simulated customer enters the system. The process is repeated until no further improvement of the objective function can be obtained.

Section 2 provides a review of the state of the art in the domain of online booking of mobile personnel and highlights the unique contribution of this study. In Section 3, we formulate the problem. Section 4 describes the proposed booking policy optimization method. Section 5 presents a numerical experiment that compares the performances of the optimized policy to a naïve baseline policy widely used in practice. Section 6 presents the final remarks and draws directions for future research.

2 Literature review
Online technician booking problems are characterized by the arrival of requests at different points in time according to some stochastic process and during a booking horizon. Each request is accepted or
rejected online upon arrival, following a short interaction between the customer and the service provider. The booking horizon ends at a certain cutoff time that precedes the service horizon, during which the requests are served. Deciding whether to accept or reject a request may be done solely based on the previously accepted requests, that is, myopically. In a more farsighted approach, the decision process may incorporate some information related to future requests; such a framework is referred to as anticipatory.

Closely related problems are dynamic vehicle routing problems (VRP) with stochastic requests. In these models, customer requests arrive during the service horizon and should be accepted or rejected promptly. However, in the dynamic VRP setting, there is no notion of booking horizon and cutoff time. For comprehensive surveys, the readers may refer to Pillac et al. (2013) and Psaraftis et al. (2016). For studies on dynamic routing with stochastic requests, see Bent and Van Hentenryck (2004), Ghiani et al. (2009), Ghiani et al. (2012), Soeffker et al. (2017), Ulmer et al. (2018b), Ulmer (2019), Voccia et al. (2019).

Several authors study tactical planning in the context of online booking. That is, the offline process of deciding on service zones, allocation of personnel to zones, and service time slots. See, for example, Agatz et al. (2011), Hernandez et al. (2017), Bühler et al. (2016), Restrepo et al. (2019), Klein et al. (2019).

The literature on optimal operation of booking systems is divided by the way in which the customer side is modeled during the booking process. Some early studies assume that the probability of accepting each offer is externally known or that the customer specifies her possible time slots, and the provider selects one of them. Most recent studies characterize customer behavior using a discrete choice model, such as the MNL and assume the model’s parameters are known to the provider.

Another important distinction is the method by which the demand is controlled in order to maximize the system profits or service level. The most prevalent approach is to set the price of the service at each slot dynamically. An alternative approach, which fits various service situations, is by availability control, i.e., the provider presents the customer with a limited set of choices (time slots). The optimization challenge here is to solve an assortment problem upon the arrival of each request. For reviews of choice models and their application in booking systems, see Strauss et al. (2018) and Klein et al. (2020).

Campbell and Savelsbergh (2005) studied a problem in which the information related to the potential delivery requests is known at the beginning of the booking horizon. The customer presents a set of feasible time slots, and the provider selects one of them or rejects the customer. The objective function is to maximize the total revenue from serving requests net of the routing costs. The authors present various myopic and anticipatory decision criteria for the acceptance of newly arrived delivery requests. The numerical experiments demonstrated the merit of the anticipatory approach.

Campbell and Savelsbergh (2006) were the first to study an online booking problem where the provider presents a different price (or discount) dynamically to each slot. The customer chooses one of them based on a function that maps the slots’ prices to selection probabilities. The objective function is to maximize the total revenue net of the routing costs. The superiority of the proposed dynamic pricing policy over more straightforward policies is demonstrated.
Cleophas and Ehmke (2014) studied an online booking problem combined with capacity planning for multiple service regions. The customer presents his single possible slot, and the provider accepts it if enough capacity was pre-allocated to meet the request. Otherwise, the request is rejected. The challenge addressed by this study is to plan the capacity so as to maximize the total profits. Ehmke and Campbell (2014) studied an online booking problem and incorporated time-dependent and stochastic travel times in their booking consideration. They demonstrated that considering travel time stochasticity yields routing plans that minimize the actual lateness at the customers' locations.

Köhler et al. (2020) studied an online booking problem with availability control where the service provider presents the customer with a set of possible time slots with two possible lengths, long and short. The objective is to maximize the number of accepted customers. The authors present decision criteria related to the fleet's current utilization and the location of the delivery request and demonstrated their merits over a baseline policy that offers short time windows only.

Asdemir et al. (2009) studied an online booking problem where each request is associated with a certain revenue and capacity consumption. The provider presents the customer with a set of time slots that do not violate the capacity constraint and determines a price for each of the presented slots. The customer chooses one of the proposed time slots or abandons the process according to an MNL discrete choice model. The utility in the MNL model is determined by the presented price and each slot's particular characteristic. The authors devised a policy that determines the prices to maximize the expected total profit of the provider. The problem was formulated as a Markov decision process (MDP) that identifies the opportunity cost of serving a request at each slot. In other words, the future profit obtained if the newly arrived request is not served net of the future profit obtained if it is. The intricate interrelations among the remaining time until the cutoff, prices, and opportunity costs were examined in this work. However, the routing aspects are not considered.

Yang et al. (2016) studied a similar problem to Asdemir et al. (2009) but in a more general setting where each request is associated with a location and the routing is considered. The price determination problem is formulated as a stochastic dynamic program. Two approximation heuristics for the opportunity cost were devised: (1) a myopic approach based on the immediate marginal cost; (2) an anticipatory approach in which the opportunity cost is obtained as a dynamic weighted average of the immediate marginal cost and an estimation of the marginal cost that is based on historical final routing plans. Moreover, Yang et al. (2016) showed the applicability of estimating the MNL model from historical data. Finally, the numerical experiment demonstrated the merit of dynamic pricing over static pricing and the superiority of the anticipatory approximation method over the myopic one. Yang and Strauss (2017) solved a similar problem using an offline approximate dynamic programming (ADP) framework to calculate the prices. To cope with the underlying routing problem, the delivery region is decomposed into smaller zones. Each zone is served by a single vehicle, and the travel distance is approximated as a linear function of the number of accepted delivery requests associated with the zone.

Koch and Klein (2020) studied an online home delivery booking problem with dynamic pricing in which customers follow the reservation price choice model, which is a generalization of the MNL. The objective function is to maximize the sum of gross profits plus the fees for the deliveries net off the routing cost. The opportunity cost of each slot is estimated using a linear value function approximation (VFA) technique. Dummy customers, representing future expected customers, are initially included in the routing plan for a more accurate estimation of the opportunity cost. The numerical experiments
demonstrated the superiority of the presented VFA estimations of the opportunity cost compared with simpler baseline pricing policies as well as the merits of anticipating future requests using dummy customers.

Strauss et al. (2020) studied an online booking problem in which the customer can express their possible degree of flexibility regarding the service's desired time. The customer's choice model is the nested MNL. Upon the arrival of a request, the customer is presented with a list of bundles of time slots, and each bundle is associated with a fee. Each bundle may contain one or several slots. The customer may choose one of the offered bundles or abandon the system. The service will be provided at one of the slots in the selected bundle at the firm's discretion and without a commitment to the customer in advance. The problem is to dynamically price the bundles with the objective of maximizing the expected gross profit plus the fees net of the routing costs. Large bundles tend to be cheaper than ones with fewer slots. A dynamic stochastic formulation of the problem is presented. The bundles' opportunity costs are obtained using a novel linear programming model that considers future delivery requests and their approximated cost. The numerical experiments demonstrated that letting the customers select bundles with more than one time slots significantly improves the firm's profit.

Mackert (2019) studied an e-grocer delivery booking problem with availability control. The customers are offered a list of delivery slots to choose from with the goal of maximizing the retailer profit net of the delivery cost by selecting the best assortment of time slots. The decisions made by the customers (choose one of the slots or abandon) are modeled by the generalized attraction model (GAM) proposed by Gallego et al. (2015). The author advocates the GAM instead of its more commonly used special case MNL since it allows circumventing an overestimation of the demand by reflecting customers' 'dissatisfaction' of small assortments of offered slots. The approximation of each request's opportunity cost is inspired by a method presented by Klein et al. (2018). A mathematical program is solved to find an optimal assortment of time slots to offer. Mackert (2019) studies a model that is similar to ours. However, it considers booking for a single period while in the current study, the service horizon consists of several periods, and the goal is to maximize the ratio of accepted requests in the long run.

Ulmer and Thomas (2019) studied an online booking problem in which each delivery request is associated with revenue, location, and vehicle capacity consumption. A request is either served at any time in the service horizon or rejected. The objective function is to maximize the sum of the revenues. The authors apply an ADP framework and devise a new VFA procedure that combines parametric and non-parametric VFA methods.

This paper is the first to study a multiday booking problem, in which requests are accepted for several working days in advance, where each day has its own cutoff time. The customers are presented with several slots that may extend over several working days to choose from. The service horizon is rolled forward by one day at the beginning of each working day. In contrast to the single cutoff case, if booking is allowed several days in advance and each day has its own cutoff time, there are inherent interactions between the various days. Such interactions call for optimizing the steady-state performance of the booking system rather than optimizing each period separately. This paper introduces an effective policy optimization method and performs a numerical experiment that demonstrates its merits. Moreover, we demonstrate that additional flexibility obtained from the multiday setting results in a more efficient operation of the service crew.
In Table 1, we list the surveyed papers on optimal operation of booking systems and describe their characteristics in terms of routing decisions, demand control method, customer choice modeling, length of the service horizon, and farsightedness of the solution method. The last row refers to the current study and demonstrates the gap that it closes in the literature.

**Table 1: Comparison of existing literature on online booking problems**

<table>
<thead>
<tr>
<th>Paper</th>
<th>Routing decisions</th>
<th>Demand control</th>
<th>Customer choice model</th>
<th>Service horizon</th>
<th>Anticipatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Campbell and Savelsbergh (2005)</td>
<td>Yes</td>
<td>-</td>
<td>-</td>
<td>Single Period</td>
<td>Yes</td>
</tr>
<tr>
<td>Campbell and Savelsbergh (2006)</td>
<td>Yes</td>
<td>Pricing</td>
<td>The provider knows the probability of the customer accepting each possible offer.</td>
<td>Single Period</td>
<td>-</td>
</tr>
<tr>
<td>Cleophas and Ehmke (2014)</td>
<td>Yes</td>
<td>-</td>
<td>-</td>
<td>Single Period</td>
<td>Yes (at the capacity planning phase)</td>
</tr>
<tr>
<td>Ehmke and Campbell (2014)</td>
<td>Yes. Time dependency and stochasticity in travel times</td>
<td>-</td>
<td>-</td>
<td>Single Period</td>
<td>-</td>
</tr>
<tr>
<td>Köhler et al. (2020)</td>
<td>Yes</td>
<td>Availability + Flexibility</td>
<td>Based on the availability time of the customer and preferences for shorter delivery slots.</td>
<td>Single Period</td>
<td>-</td>
</tr>
<tr>
<td>Asdemir et al. (2009)</td>
<td>-</td>
<td>Pricing</td>
<td>MNL</td>
<td>Single Period</td>
<td>Yes</td>
</tr>
<tr>
<td>Yang et al. (2016)</td>
<td>Yes</td>
<td>Pricing</td>
<td>MNL</td>
<td>Single Period</td>
<td>Yes</td>
</tr>
<tr>
<td>Yang and Strauss (2017)</td>
<td>Approximated based on the number of accepted requests in each zone</td>
<td>Pricing</td>
<td>MNL</td>
<td>Single Period</td>
<td>Yes</td>
</tr>
<tr>
<td>Koch and Klein (2020)</td>
<td>Yes</td>
<td>Pricing</td>
<td>Reservation price choice model</td>
<td>Single Period</td>
<td>Yes</td>
</tr>
<tr>
<td>Strauss et al. (2020)</td>
<td>Approximated based on the number of accepted</td>
<td>Pricing + Flexibility</td>
<td>Nested MNL.</td>
<td>Single Period</td>
<td>Yes</td>
</tr>
</tbody>
</table>
requests in each zone | Availability | GAM | Single Period | Yes |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mackert (2019)</td>
<td>Yes</td>
<td>-</td>
<td>Single Period</td>
<td>Yes</td>
</tr>
<tr>
<td>Ulmer and Thomas (2019)</td>
<td>Yes</td>
<td>-</td>
<td>Single Period</td>
<td>Yes</td>
</tr>
<tr>
<td>This paper</td>
<td>Yes</td>
<td>Arbitrary discrete choice model (numerical experiment with MNL)</td>
<td>Multi-period. Optimize long run acceptance ratio</td>
<td>Yes</td>
</tr>
</tbody>
</table>

3 Problem definition

The online steady-state technician booking problem is stated as follows: service requests arrive according to a non-homogeneous Poisson process over an infinite horizon. Each request is characterized by its location, service time, and particular preferences regarding the service's starting time. The infinite horizon is divided into working days, and each day is divided into several non-overlapping service windows (e.g., 8–10, 10–12, 12–14, 14–16, 16–18). A combination of a service day and a window is referred to as a slot. Upon the arrival of a request during day $h$, it can be either rejected or scheduled to a slot in one of the following days $\{h + 1, h + 2, \ldots, h + HP\}$; This period is referred to as the service horizon. From the description above, it follows that the cutoff time associated with day $h + 1$ is the end of day $h$.

The accepted requests are served by a crew of $M$ homogenous mobile technicians available on each working day. The technicians depart from a depot, not before a predefined time $a_0$, and have to return to the depot no later than $b_0$. In other words, the working day duration is $L = b_0 - a_0$. The travel time between any two locations $j$ and $k$ in the service area, $t_{jk}$, is known and deterministic.

Each newly arrived service request is initiated by a customer and triggers the execution of the following protocol.

- The system displays an assortment of time slots in the service horizon for the service's possible starting times.
- The customer may select one of the proposed time slots or abort the process without booking his request.
- Once the customer selects a slot, the service request is booked in the system and inserted into the selected time slot's tentative schedule.

The assortment is a subset of the feasible slots in which the request may begin. A slot is considered feasible if assigning the request to that slot causes no late arrivals at the previously accepted service requests and allows the technicians to return to the depot by the end of the working day. The assortment may be empty, in which case, the request is rejected immediately.

It should be noted that the actual schedules and routes of the technicians for each working day can be reoptimized after the cutoff time by solving a vehicle routing problem with time windows (VRP-TW). Moreover, in a typical situation, there are several hours between the cutoff time (say at midnight)
and the beginning of the working day (say at 7 am). During these hours, the system can exert additional computation resources to achieve a plan with shorter total travel time compared to the original tentative one.

A solution for the O-SSTBP is a policy for offering time slots to customers with the objective of maximizing the expected number of accepted service requests per day in the long term.

The description of a discrete choice model by which the customers select from the offered time slots or decide to abandon the system is included in the input of the O-SSTBP. We assume that all the customers follow the same choice model and require the same service time denoted by $\tau$. These assumptions may be applicable if the service provider has little information regarding the requested service or if the service is uniform by nature, e.g., home delivery. The set of the time slots offered to a customer is denoted by $A$. In addition, we denote the alternative of abandoning the system by $w_0$. That is, the set of alternatives available to the customer is $\{w_0\} \cup A$. In the numerical section of this paper, we use an MNL model as an example; however, any discrete choice model could be used.

We formulate the problem of maximizing the expected number of accepted requests per day at a steady-state as a Markov decision process (MDP) with an infinite horizon. For this end, we define the following notation

- $\mathcal{A}$: The set of time slots in the service horizon. Each subset of slots (an assortment) can be a feasible action at a particular state.
- $\mathcal{L}$: Set of locations of all the potential customers. For simplicity of presentation, we assume here that this is a discrete set.
- $T$: The number of time intervals during a single day. Each interval is short enough to neglect the probability that the number of arriving requests during the interval is greater than 1. The intervals are indexed as $t = 1, \ldots, T$.
- $\lambda_t$: The arrival probability of a request during interval $T$.
- $p_{lt}$: The probability that a request from location $l \in \mathcal{L}$ arrives at time interval $t \in T$. Note that $\sum p_{lt} = \lambda_t$.
- $P(A, w)$: The probability that a customer will choose time slot $w$ when presented with an assortment of time slots $A \subset \mathcal{A}$. Recall that the alternative of abandoning the system is denoted as $w_0$. Therefore, $P(A, w_0)$ denotes the probability of not accepting any slot when the assortment $A$ is presented. We use the convention that $P(\emptyset, w_0) = 1$. Note that $P(A, w) = 0$ if $w \notin A$.
- $S_t$: The state of the system at time interval $t$. A state includes information about the number of accepted customer requests at each particular time slot in the service horizon, $w \in \mathcal{A}$, each location, $l \in \mathcal{L}$, and the time of the day $t = 1, \ldots, T$.
- $\mathcal{S}$: The set of all the feasible states.
- $\pi(S_t, l)$: A policy $\pi$ is a mapping of $\pi: (S_t, l) \rightarrow 2^{\mathcal{A}}$. The policy prescribes a set of feasible slots to offer to a customer from location $l$ that places a service request at time $t$ of the day while the state of the system is $S_t$. $2^{\mathcal{A}}$ denotes the collection of all subsets of time slots in the service horizon. We use the convention that $\pi(S_t, l_0) = \emptyset$ to refer to situations where no request arrives at time $t$.
- $\Pi$: The set of all policies.
\[ N(S_t) \quad \text{For } t < T, \text{the set of states at time } t + 1 \text{ that the system can reach from state } s_T \text{ as a result of booking exactly one customer request. For the last interval in a day } N(S_T) \text{ consist of the states } S_1 \text{ that can be obtained from accepting an additional one request in a slot that does not belong to the service horizon's first day. Note that these states refer to the new booking horizon after the cutoff time at the end of interval } T. \text{ I.e., the slots of the first day are removed, the rest of the slots are advanced by one day, and new empty slots for the last day are added.} \]

\[ I(S_t) \quad \text{For } t < T, \text{the state at time interval } t + 1, \text{if no request was accepted at time } t. \text{ For the last interval } I(S_T) \text{ is the state } S_1 \text{ if no request was accepted at time } T \text{ or a request was accepted for the first day of the service horizon.} \]

\[ \Delta(S_t, S_{t+1}) \quad \text{For } S_{t+1} \in N(S_t) \text{ and } t < T, \text{the location of the accepted request at time } t. \text{ For } S_1 \in N(S_T), \text{ the location of the accepted request at time } T. \]

\[ W_t \quad \text{The set of slots on the first day of the service horizon} \]

Using this notation, we can define the transition probability between any pair of states in consecutive time intervals, given a request at location \( l \) for \( t = 1, \ldots, T - 1 \), as follow.

\[
P(S_t, S_{t+1}) = \begin{cases} 1 - \lambda_t + \sum_{l \in L} p_{lt} \mathcal{P}(\pi(S_t, l), w_0), & S_{t+1} = I(S_t) \quad S_{t+1} \in N(S_T) \\ 0, & \text{otherwise} \end{cases} \tag{1}
\]

The first case refers to transitions resulting from progressing from time interval \( t \) to \( t + 1 \) without accepting any request. Note that by \( S_{t+1} = I(S_t) \) indicates that the only difference between the states \( S_t \) and \( S_{t+1} \) is the time. The probability of this transition is the sum of two probabilities of two mutually exclusive cases. Either no customer request arrive during interval \( t \) (with a probability of \( 1 - \lambda_t \)) or a request arrives, an assortment \( \pi(S_t, l) \), is offered by the operator and rejected by the customer. The second case refers to transitions resulting from progressing from time interval \( t \) to \( t + 1 \) where a request is accepted. The location of the accepted request is encoded in the information about states \( S_t \) and \( S_{t+1} \) as \( \Delta(S_t, S_{t+1}) \) and the slot as \( w(S_t, S_{t+1}) \). The third case refers to all the other transitions, which are infeasible in a one-time interval. For time interval \( T \), just before the cutoff time, the transition probabilities are

\[
P(S_T, S_1) = \begin{cases} 1 - \lambda_t + \sum_{l \in L} p_{lt} \sum_{w \in W_T(w_0)} \mathcal{P}(\pi(S_t, l), w), & S_1 = I(S_T) \\ p_{\Delta(S_T, S_1), t} \mathcal{P}(\pi(S_t, \Delta(S_T, S_1)), w(S_T, S_1)), & S_1 \in N(S_T) \\ 0, & \text{otherwise} \end{cases} \tag{2}
\]

The first case refers to transitions from time interval \( T \) to \( 1 \) without accepting any request or when the accepted request is for the first day of the service horizon and hence will be deleted from the state at the cutoff time at the end of time interval \( T \). The first term is as in (1), and the second term is a sum over the events where the customer either chooses to abandon or chooses a slot on the first day of the service horizon. The second and third cases are exactly as in (1) but note the unique definitions of \( N(S_T), \Delta(S_T, S_1), \) and \( w(S_T, S_1) \).
When a policy $\pi$ is known, the transition probability above defines a Markov chain over the states $S_t$. We denote the steady-state probability of this chain at time interval $t$ by $Q_t(S_t, \pi)$.

An optimal solution is a policy that satisfies

$$\pi^* = \arg\max_{\pi \in \Pi} \sum_{t=1}^{T} \sum_{S_t \in S} Q_t(S_t, \pi) \cdot \sum_{l \in \mathcal{L}} p_{lt} \sum_{w \in \pi(S_t, l)} P(\pi(S_t, l), w)$$

Note that the expected number of accepted requests per day in steady-state is calculated as the sum of probabilities of accepting a request at each time interval. For each particular state $S_t$ the probability of accepting a request at the corresponding interval is calculated by the sum of probability of the arrival events multiplied by the probability one of the offered slots will be accepted.

Unlike previous authors in the booking literature, we used a general MDP rather than a sequential set of Bellman equations to define our problem formally. The MDP framework is suitable because we are interested in the expected number of accepted requests in a steady-state rather than the number of accepted requests during a finite booking horizon.

4 Methodology

Implementation of the MDP presented in Section 3 is not practical for realistic settings since the sets of states and actions (assortments) are too large. Instead, we encode each state and the information about the new request in a compact way and use simulation results to fit a Cobb–Douglas function that approximates the opportunity cost of accepting a request from each possible location at each time slot in the service horizon. Upon the arrival of a service request, an assortment of feasible slots that approximately maximizes the expected gain from the customer is constructed and offered to the customer. Such an assortment is obtained by selecting one that maximizes the immediate (unit) gain net of the approximated opportunity cost. At any particular moment, the booking system maintains a tentative set of routes and schedules for all the technicians during the entire service horizon. Once a customer selects a time slot, his request is inserted into one of the tentative routes, and the schedule is updated. The tentative plan and the insertion procedure enable us to quickly evaluate the feasibility of inserting the request at each slot.

The idea of estimating the opportunity cost is well established in the booking literature, as discussed in Section 2. However, we present the first application of this idea for optimizing the long-run performances of a booking process rather than focus on a single period with a single cutoff.

Section 4.1 presents the approximated opportunity cost function and explains the modeling considerations. Section 4.2 presents the routing and scheduling oracle that we implement to aid in the cost estimation for each slot as well as for the tentative scheduling of the accepted requests. Section 4.3 presents the optimization model used to construct the list of offered slots. Finally, Section 4.4 presents the proposed procedure for fitting the parameters of the opportunity cost function.

4.1 The approximated opportunity cost function

The approximated opportunity cost function estimates the implications of accepting a particular customer request at a slot, in terms of the number of requests that will either be rejected or will abandon the system due to the resources being allocated to the current request.
Consider a route $r'$ that represents the set of requests that a technician visits in a given day, as well as the starting times at each customer location and the departure and arrival time at the depot. Let $i$ be a newly arrived request that is a candidate to be added to route $r'$ at time slot $s$. A new route $r$ that consists of all the requests in $r'$ as well as request $i$ is said to be feasible if the slots of all the previous requests in $r'$ are not changed, and the service of request $i$ starts at time slot $s$. Based on $(r, r', i, s)$, one can calculate the following three measures:

\[
\begin{align*}
RTS(r, s) & \quad \text{Remaining idle time in slot } s \text{ (after the insertion of } i \text{ into the route at slot } s). \text{ The technician is considered idle when no service or travel is carried out.} \\
RTR(r, s) & \quad \text{Remaining idle time in route } r \text{ during the entire working day (after the insertion of } i \text{ into route } r \text{ at slot } s). \\
TT(r', r) & \quad \text{Increase in the travel time incurred by extending } r' \text{ to } r.
\end{align*}
\]

The calculation of $RTS$, $RTR$, and $TT$ is discussed in detail in Section 4.2. Note that the values of the above three measures are nonnegative. Next,

\[
f(r, r', s) = \begin{cases} 
(RTS(r, s) + RTR(r, s) + TT(r', r))^\gamma, & \text{if route } r \text{ is feasible} \\
0, & \text{otherwise}
\end{cases}
\]  

where $\alpha, \beta, \gamma$ are parameters that characterize the opportunity cost function $f$ and may be either positive or negative. The Cobb-Douglas function $f(r, r', s)$ represents the expected number of future requests that will be lost if the system accepts the current request in route $r$ at slot $s$. Therefore, $f(r, r', s) > 1$ implies that serving the current request in route $r$ at slot $s$ is not worthwhile. The Cobb-Douglas function is often used by economists to represent production and utility with respect to a given amount of inputs. We used it to approximate the opportunity cost because it is one of the simplest functional forms that can capture various resources' interactions.

The measures $RTS$, $RTR$, and $TT$ capture some information about the system's state in a concise manner that is sufficient to estimate the opportunity cost effectively. The case for $TT$ is straightforward: a higher marginal increase in the travel time corresponds to a higher likelihood that the future requests will be lost. Thus, $\gamma$ is likely to be positive. Next, high values of $RTS$ and $RTR$ indicate that the candidate route (and in the case of $RTS$, the candidate slot as well) is sufficiently vacant to serve additional requests after accepting request $i$. Thus, accepting $i$ is unlikely to adversely affect the offers that will be made to future customer requests. Hence, $\alpha$ and $\beta$ are likely to be negative. We note that Ulmer et al. (2018a) and Koch and Klein (2020) use similar measures to aggregate the state space in their ADP frameworks. In Section 4.4, we present a method for fitting appropriate values of $\alpha, \beta, \gamma$.

Note that $RTS, RTR,$ and $TT$ may be zero. In this case, since the exponent may be negative, to ensure that the Cobb–Douglas cost function is defined, we correct (3) by adding a small $\varepsilon > 0$ to each of the measures. Specifically,

\[
f(r, r', s) = \begin{cases} 
(RTS(r, s) + \varepsilon)^\alpha(RTR(r, s) + \varepsilon)^\beta(TT(r', r) + \varepsilon)^\gamma, & \text{if route } r \text{ is feasible} \\
0, & \text{otherwise}
\end{cases}
\]  

(3')

\[\text{11}\]
4.2 The routing and scheduling oracle

This section presents a procedure to insert new requests to the existing plan at particular routes and time slots. This procedure is used to construct the tentative routes and schedules and approximate the corresponding opportunity costs. These approximate values are later used to construct assortments offered to the customer, as described in Section 4.3. The opportunity cost of scheduling a request at a time slot is defined as the opportunity cost of assigning the request to the technician with the minimal opportunity cost at the time slot. We refer to this value as the opportunity cost of the time slot, for short.

In Figure 1, we present a pseudocode of a program that finds the set of time slots with opportunity costs smaller than 1. For each such time slot, the program returns the technician for which the request should be tentatively assigned, the tentative position of the request in his route, and the approximate opportunity cost of the time slot. It should be noted that the routing and scheduling problem is not the focus of this study, and any other fast routing procedure can replace the proposed oracle.

Input:
Set of service time slots at the next $H_p$ working days, denoted by $S$
Set of technicians, denoted by $K$
Tentative routes of all the technicians in the next $H_p$ working days
Parameters of the opportunity cost function $\alpha, \beta, \gamma$
Identification of the current request, $i$, and its properties

for $s \in S$
  $C[s].f \leftarrow \infty$
for $k \in K$
  $r' \leftarrow$ current route of $k$ in the day of slot $s$
  for $p = 0$ to $|r'|$ \hspace{1cm} // where $|r'|$ is the number of customers in route $r'$
    $r \leftarrow$ route of $k$ after inserting request $i$ after position $p$ to $r'$
    if route $r$ is feasible for slot $s$
      calculate $RTS(r,s), RTR(r,s), TT(r',r), f(r,r',s)$
      if $f(r,r',s) < C[s].f$
        $C[s].tech \leftarrow k$
        $C[s].f \leftarrow f(r,r',s)$
        $C[s].route \leftarrow r$
  $S' = \{s \in S | C[s].f \leq 1\}$
Return: $S', C$

Figure 1: Pseudocode for finding a set of candidate slots

We define, for each slot $s$, an object $C[s]$ that stores information regarding the best-found route to serve request $i$ at slot $s$. In particular, $C[s].tech$ is the identity of the most appropriate technician and $C[s].route$ is the route of this technician, including the newly arrived request $i$. Recall that a route is
defined by a set of requests and their service times. In $C[s], f$, we store the approximated opportunity cost for the particular request and slot. We initialize $C[s], f$ to infinity.

Our algorithm updates $C$ by looping through all the slots, technicians, and insertion positions along the routes. Note that inserting a request at position $p$ means that the request is inserted immediately after the $p^{th}$ customer in the sequence, with the convention that insertion at position 0 means inserting at the first position in the route. After the insertion, the starting times of all requests starting from the position of the inserted one are set to the earliest possible times in their pre-committed slots. Recall that the obtained route is said to be feasible if all its requests start at their required time slots.

The technician's route and schedule with the minimal opportunity cost are found for each slot. Finally, a list with all the slots with an expected opportunity cost that is not greater than one, $S'$, as well as the best-found technicians, routes, and their opportunity costs (stored in $C$) are returned. A subset of the slots in this list is offered to the customer, as discussed in Section 4.3.

Next, we elaborate on the calculation of $RTS$, $RTR$, and $TT$. To calculate $TT$, let $j$ and $k$ denote the previous and next customers of $i$ in $r$, respectively. Thus, $k$ follows $j$ in route $r'$. Now, $TT = t_{ji} + t_{ik} - t_{jk}$.

Let $u_j$ denote the service starting time at the location of request $j$, and let $a_s$ and $b_s$ denote the starting and ending times of slot $s$, respectively. The routing and scheduling oracle inserts all the customers at the earliest possible starting time $u_j$ that maintains their time slots constraints. Specifically, if customer $k$ is the immediate successor of customer $j$, and $k$ is assigned to slot $s, u_k = \max(a_s, u_j + \tau + t_{jk})$ where $\tau$ is the service time. To calculate the $RTS$ for slot $s$, let $j$ denote the customer whose service starting time is the latest in slot $s$. That is, $j = \arg\max_{j \in R} \{u_j | a_s \leq u_j \leq b_s\}$, where $R$ is the set of customers in route $r$. Next, let $\hat{k}$ denote the immediate successor of $j$. Hence,

$$RTS = b_s - \min(u_j + \tau + t_{jk}, b_s).$$

After serving customer $j$, the technician is ready to start the service of $\hat{k}$ at time $u_j + \tau + t_{jk}$. If this time is still within slot $s$, i.e., before $b_s$, the remaining time, $RTS$, is $b_s - (u_j + \tau + t_{jk})$; otherwise, it is zero.

Finally, the $RTR$, is obtained by the difference between the length of the working day, $L$, and the total travel and service time allocated to the technician.

### 4.3 Constructing an assortment

After calculating the approximated opportunity cost of scheduling the request at each slot, the planner can construct an assortment of slots to offer that will approximately maximize the expected number of current and future requests that will be served. Recall that the net gain from serving a customer at a particular slot is one minus the expected number of future requests that will be lost due to the resources allocated for the current request. The problem of selecting a subset of slots from the candidate slot list, $S'$, which is created as described in Section 4.2 to maximize the expected net gain, can be formulated as a nonlinear mixed-integer program (2)–(4). For each $s \in S'$, we define a binary decision $x_s$ that indicates whether the slot is included in the assortment and an auxiliary variable $P^s$ that holds the probability that the customer will select slot $s$. 
\[
\max \sum_{s \in S'} P^s (1 - C[s].f) \\
\text{subject to}
\]
\[
P^s = g_s(x) \quad \forall s \in S' \tag{5}
\]
\[
x_s \in \{0,1\} \quad \forall s \in S' \tag{6}
\]

The objective function maximizes the expected net gain from the offer by weighting the expected gain from each slot with the probability that the customer will select it. Constraint (5) relates the auxiliary variables \(P^s\) to the offered slots represented by \(x\). The function \(g_s(x)\) represents the discrete choice model and calculates the probability of selecting slot \(s\) when the assortment is represented by the characteristic vector \(x\). We also comment that

1. For a slot \(s\) that is not included in the offer (i.e., \(x_s = 0\)), \(P^s = 0\).
2. The sum \(\sum_{s \in S'} P^s\) \(\leq 1\). The inequality is strict if the probability of a customer abandonment is positive.

If the choice model is MNL, (3) can be replaced by

\[
g_s(x) = \frac{e^{u_s x_s}}{\sum_{q \in S'} e^{u_q x_q} + 1} \quad \forall s \in S' \tag{5'}
\]

In the denominator, we add the term 1 to represent the abandonment alternative. Note that the utility to a customer from abandoning the system is 0, and \(e^0 = 1\).

In the numerical experiment, the optimization problem (4)–(6) is solved by enumerating all the subsets of \(S'\). A service horizon of 15 slots (three days and five slots per day) is tested; therefore, the number of possible subsets is not greater than \(2^{15}\). However, the number of candidate slots in \(S'\) is typically considerably smaller than 15, and thus, the problem is solved quickly. We note that this enumerative approach is applicable to any computable choice model and not limited to the MNL used for demonstration in our experiment. For longer service horizons with many slots, Mackert (2019) presented a linear programming model for solving the assortment problem related to the choice model of GAM, which is a generalization of the MNL model. However, solving the assortment model for other choice models may be NP-hard. In these cases, one may either use enumeration of all the possible assortments or use heuristic methods.

Once the optimal assortment is presented to the customer, he may either select one of the slots or abandon the system. If the customer selects a slot \(s\), the request is inserted in the route of technician \(C[s].tech\), as prescribed by the algorithm presented in Figure 1. The resulting plan is tentative, and the actual routes and schedule may be reoptimized later.

### 4.4 Fitting parameters for the opportunity cost function

In this section, we present a procedure for fitting values for the opportunity cost model parameters. For any combination of the parameters \(\alpha, \beta,\) and \(\gamma\), the opportunity cost function \(f\) defines a policy with (possibly) different expected number of accepted requests per day. The objective is to determine the model parameters that maximize this number. The procedure described below converges to a local
maximum of this optimization problem, but convergence to a global optimum cannot be ensured. In Section 5, we provide numerical evidence for the effectiveness of the proposed heuristic.

The proposed procedure relies on a variant of the steepest descent method. The procedure has three phases that are applied repeatedly: 1) estimation of the steady-state performance of a current solution; 2) estimation of directional derivatives for the three parameters; 3) improving the current solution by applying a line search in the most promising direction. The process is repeated until a stopping criterion is met.

At Phase 1, we estimate the expected number of accepted requests for a given set of parameters using a discrete event simulation. Specifically, we simulate multiple working days and calculate an estimator for the expected number of accepted requests at steady-state by ignoring the first few (warmup) days. The same stream of orders is used at all the iterations to reduce the variability, and each iteration consists of one very long period multiple times.

At Phase 2, we estimate the expected number of accepted requests for 26 neighboring combinations of $\alpha, \beta, \gamma$. In our framework, the neighborhood of a current solution $x^{(n)} = (\alpha^{(n)}, \beta^{(n)}, \gamma^{(n)})$, denoted by $\mathcal{N}$, is obtained as the set of all the solutions generated from increasing, fixing, or decreasing the value of the three parameters by a small nonnegative constant $\Delta$. The neighborhood is denoted as

$$\mathcal{N}(\alpha^{(n)}, \beta^{(n)}, \gamma^{(n)}) = \{(\alpha^{(n)}, \alpha^{(n)} + \Delta, \alpha^{(n)} - \Delta) \times (\beta^{(n)}, \beta^{(n)} + \Delta, \beta^{(n)} - \Delta) \times (\gamma^{(n)}, \gamma^{(n)} + \Delta, \gamma^{(n)} - \Delta\}$$

The expected number of accepted requests of each solution in $x \in \mathcal{N}$ is evaluated by simulation.

At Phase 3, if one or more of the neighboring solutions is better than $x^{(n)}$, let $x^{(n')} \in \mathcal{N}$ be the best solution in $\mathcal{N}$ and thus $x^{(n')} - x^{(n)}$ is the best improving direction. Next, we initiate a line search along a segment of the ray from $x^{(n)}$ at the direction of $x^{(n')}$, that is, in

$$\mathcal{M} = \{((\alpha, \beta, \gamma)|x^{(n)} + l(x^{(n')} - x^{(n)}), \ l \in \mathbb{R}_+\}$$

We use the golden section method to find a locally optimal solution $x^{(n+1)}$ in the ray $\mathcal{M}$. Note that since $x^{(n')} \in \mathcal{M}$, the best solution obtained by the line search is at least as good as $x^{(n')}$ and thus strictly better than $x^{(n)}$.

If no improving solutions are found in $\mathcal{N}$, $x^{(n+1)} \leftarrow x^{(n)}$, and $\Delta$ is updated as $\Delta \leftarrow \Delta/\omega$, where $\omega > 1$ represents an update factor. Note that reducing $\Delta$ redefines the neighborhood with solutions that are more similar to the current one, which can be beneficial in later stages of the search, in which the current solution is already close to a local optimum.

The search procedure starts from some initial solution $x^{(0)} = (\alpha^{(0)}, \beta^{(0)}, \gamma^{(0)})$ and iterates to generate a series of solutions with nondecreasing values. The value of $\Delta$ is reset to its initial value after each improvement. If no improving solution is found for a predefined number of iterations, the search process is terminated.
5 Numerical experiments
In this section, we present the numerical experiments performed to benchmark the proposed method and derive some insights. Section 5.1 describes the benchmark problem instances. Section 5.2 describes the performance analysis for the proposed method and reports the results. In Section 5.3, we demonstrate the merits of a long service horizon with multiple cutoff times.

Since this is the first study to consider optimal booking in a multiday setting with the objective of maximizing the steady-state acceptance rate, it is not possible to compare the performance of our methods to previously presented ones. Therefore, we use a myopic policy that offers all the available slots to each arriving request as a baseline. We believe that our baseline policy is similar to the approaches that are widely used in practice.

5.1 Description of problem instances
This section describes the settings of the benchmark instances used in the main experiment reported in Section 5.2. Some of the parameter values are diversified in Section 5.3 to gain additional insights.

The service crew in our benchmark instances consists of six technicians that are available at each working day. The working day begins at 8:00 and ends at 18:00; thus, \( L = 600 \) minutes. Each working day is divided into five non-overlapping service windows of two hours each, i.e., 8:00 – 10:00, 10:00 – 12:00, 12:00 – 14:00, 14:00 – 16:00, and 16:00 – 18:00. All the slots of the same working day have the same cutoff time, which is the beginning of the corresponding working day.

The length of the service horizon is three days, \( H^p = 3 \). Therefore, 15 potential time slots may be available for the service of a newly arrived request. The customers are assumed to follow the MNL model. The expected utilities of obtaining the service in each of the five slots on the next day, i.e., the first day of the service horizon, are presented in Table 2.

<table>
<thead>
<tr>
<th>Next Day</th>
<th>8:00 – 10:00</th>
<th>10:00 – 12:00</th>
<th>12:00 – 14:00</th>
<th>14:00 – 16:00</th>
<th>16:00 – 18:00</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

Each additional day of delay in the service results in a 20% reduction of the utility associated with each slot. The utility obtained from abandoning the process is 0. Moreover, the expected utilities are assumed to be the same for all the customers (see Section 3). This preference model reflects the fact that the customers desire to be served as soon as possible and that the time slots at the beginning or the end of the working day are more convenient than the mid-day ones.

In our experiments, the booking system is fully automated (e.g., via a website), and the service requests may arrive at any hour during the day. The arrival process of the requests to the system is assumed to be periodic (over a period of a day), and the fraction of the daily arrivals during each hour of the day, denoted by \( d_t \), is presented in Figure 2. Note that the arrival process reflects two peaks, one peak in the late morning, and the other peak in the early evening. Naturally, the arrival rate is considerably low during the nighttime.
In our problem instances, we consider three values of the expected number of requests per day, denoted by $D$, namely, 64, 72, and 80 requests. The service time $\tau$ related to each instance is determined such that the expected amount of total required service time is identical among all the instances and equals $\frac{2}{3}$ of the six technicians' total working time, i.e., 2,400 minutes, which represents 400 minutes to each of the technicians. For example, for instances with 80 customers, the service time per customer is $\tau = \frac{2400}{80} = 30$ minutes.

In all the problem instances, the customers' locations are scattered on a $100 \times 100$ square, and the depot is located at the center. We generated three types of spatial configurations, namely, urban, suburban, and rural. For the urban and suburban configuration, 8 points are randomly generated from a uniform distribution, i.e., $x_z \sim U(10,90), y_z \sim U(10,90), z = 1 \ldots 8$. Each point constitutes the center of a zone. For the rural configuration, 72 points are generated to represent the centers of zones. In this case, $x_z \sim U(2,98), y_z \sim U(2,98), z = 1 \ldots 72$. The locations of the settlement centers are kept constant throughout the experiment.

Upon the arrival of each request, its location is generated in two steps: first, we randomly pick a zone, $z$, for it, and then we generate its exact location using random polar coordinates. An angle $\theta$ and distance $r$ are randomly generated using a uniform distribution $\theta \sim U(0,2\pi), r \sim U(0,R)$ where $R = 5, 10, \text{and } 2$ for the urban, suburban, and rural configurations, respectively. Next, based on the polar coordinates, $(\theta, r)$, the $x$ coordinate is obtained as $x = x_z + r \cos(\theta)$, and the $y$ coordinate is obtained as $y = y_z + r \sin(\theta)$. The travel times between the locations are the Euclidean distances.

The urban and suburban configurations are created to reflect a setting with a small number of high demand cities in which the traveling time within each city is short to moderate. The rural configuration represents a setting that is highly dispersed, and in which the expected demand at each settlement (village or small town) is low. Figure 3 demonstrates the locations of requests that arrive at a typical day for the three types of configurations.

**Figure 2:** Fraction of requests arriving at each hour of the day

![Fraction of arrivals](image-url)
An instance of the problem is determined by the spatial configuration and the demand intensity $D$ in our experiment, since all the other parameters are constant. A training and testing streams of requests are generated for each instance. The training stream is used in the process applied to fit the parameters of each instance. The same stream is used for all the parameter combinations explored by our search procedure. The testing stream is used to evaluate the performance of the booking system with the final parameters, $(\alpha, \beta, \gamma)$, fitted in the search process. The training and test streams consist of requests obtained over 600 and 6000 booking days, respectively, and include 60 warmup days that have been excluded from the estimations, to ensure that the observed measures represent the steady state. The length of the simulation and the warmup period were determined and verified by extensive statistical analysis using a blocking technique.

We generated three instances with different random zone centers and random streams of requests for each unique combination of spatial configuration (urban, suburban, rural) and value of $D$ (64, 72, 80 requests per day). There are 27 benchmark instances in total. A python script that reproduces this dataset and its detailed description are available in the electronic appendix that can be downloaded from http://www.eng.tau.ac.il/~talraviv/Publications/.

The instances are coded according to the number of requests $D$, the number of zones $Z$, the radius of the settlements $R$, and serial replication number $S$. For example, the code name D72Z8R10S2 refers to the second instance with a rate of 72 requests per day and eight different settlements with a radius of 10 each (suburban setting).

5.2 Performance evaluation of the proposed method

In this section, we compare the proposed method's performance with a baseline myopic policy and show its superiority. In addition, we apply the proposed method while excluding each of the three model components ($RTS, RTR, and TT$) to examine the importance of each one of them.

In the experiments, the search process begins with an initial solution: $\alpha^0 = 0, \beta^0 = 0, \gamma^0 = 0$. The initial $\Delta$ is set as 0.1, and it is updated by a factor of $\omega=2$. The numerical tolerance of the Cobb–Douglas function is $\varepsilon = 0.01$. The process stops after four consecutive iterations without improving the objective function.

In the electronic appendix, we provide a table with the best-obtained values of the parameters for each instance. As expected, in all the instances, the values of $\gamma$ are strictly positive, and the values of $\alpha$ and $\beta$ are strictly negative.
We use a baseline myopic policy, similar to the common practice, to benchmark our method for generating assortments. Under this policy, upon the arrival of a request, all the feasible slots are offered. Next, following the selection of a slot by the customer (using the same choice model as in our main experiment), the request is inserted so as to minimize the additional tentative routing cost. The myopic policy and proposed policy's performance were compared based on the same geographies and testing streams of requests.

It should be noted that under the assumptions of our model and any reasonable choice model, the optimal assortment in a policy that ignores future requests includes all the feasible time slots. Indeed, it is not likely that extending the assortment by additional slots will increase the probability of abandonment. In particular, in the MNL model used in our experiment, any additional slot in the assortment strictly decreases the chance that the customer will abandon. Therefore, the assortment offered by our baseline policy is optimal in the myopic sense.

Table 4 compares the proposed method's performance and the baseline myopic policy in all the 27 instances. For each of the methods, the table presents the request acceptance ratio in the long term, average travel time per accepted request, average travel and service time per day for a technician, and average total technician working time net of waiting (for the opening of time windows). The average total technician working time equals the sum of her travel and service time. The acceptance ratio is calculated based on the number of accepted requests throughout the simulated period (excluding the warmup) and the total number of requests during this period. Recall that a request may not be accepted either if the customer is offered an empty assortment or decided to abandon the system after observing her assortment.

Table 4: Performance measures of the proposed method and a myopic policy

<table>
<thead>
<tr>
<th>Instance</th>
<th>Acceptance ratio (%)</th>
<th>Travel time per request (min.)</th>
<th>Travel time per technician per day (min.)</th>
<th>Service time per technician per day (min.)</th>
<th>Working time per technician per day (min.)</th>
<th>Acceptance ratio (%)</th>
<th>Travel time per request (min.)</th>
<th>Travel time per technician per day (min.)</th>
<th>Service time per technician per day (min.)</th>
<th>Working time per technician per day (min.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D64Z8R5S1</td>
<td>92.6%</td>
<td>16.3</td>
<td>161.5</td>
<td>370.6</td>
<td>532.1</td>
<td>97.5%</td>
<td>11.7</td>
<td>122.2</td>
<td>390.1</td>
<td>512.3</td>
</tr>
<tr>
<td>D64Z8R5S2</td>
<td>92.2%</td>
<td>17.3</td>
<td>169.7</td>
<td>368.8</td>
<td>538.5</td>
<td>96.5%</td>
<td>16.3</td>
<td>167.9</td>
<td>386.2</td>
<td>554.1</td>
</tr>
<tr>
<td>D64Z8R5S3</td>
<td>91.9%</td>
<td>18.7</td>
<td>183.3</td>
<td>367.6</td>
<td>550.9</td>
<td>97.0%</td>
<td>12.7</td>
<td>131.9</td>
<td>388.3</td>
<td>520.2</td>
</tr>
<tr>
<td>D64Z8R10S1</td>
<td>92.4%</td>
<td>18.7</td>
<td>184.4</td>
<td>369.8</td>
<td>554.2</td>
<td>96.8%</td>
<td>13.9</td>
<td>143.1</td>
<td>387.2</td>
<td>530.3</td>
</tr>
<tr>
<td>D64Z8R10S2</td>
<td>91.6%</td>
<td>20.1</td>
<td>196.0</td>
<td>366.5</td>
<td>562.5</td>
<td>96.4%</td>
<td>13.9</td>
<td>142.6</td>
<td>385.8</td>
<td>528.4</td>
</tr>
<tr>
<td>D64Z8R10S3</td>
<td>90.2%</td>
<td>22.0</td>
<td>211.4</td>
<td>361.1</td>
<td>572.5</td>
<td>96.1%</td>
<td>14.5</td>
<td>149.2</td>
<td>384.6</td>
<td>533.8</td>
</tr>
<tr>
<td>D64Z72R2S1</td>
<td>80.2%</td>
<td>30.5</td>
<td>260.8</td>
<td>320.9</td>
<td>581.7</td>
<td>92.2%</td>
<td>16.6</td>
<td>163.5</td>
<td>368.9</td>
<td>532.4</td>
</tr>
<tr>
<td>D64Z72R2S2</td>
<td>80.1%</td>
<td>30.5</td>
<td>261.1</td>
<td>320.7</td>
<td>581.8</td>
<td>92.2%</td>
<td>16.5</td>
<td>162.7</td>
<td>369.1</td>
<td>531.8</td>
</tr>
<tr>
<td>D64Z72R2S3</td>
<td>84.8%</td>
<td>26.7</td>
<td>241.6</td>
<td>339.2</td>
<td>580.8</td>
<td>93.9%</td>
<td>15.3</td>
<td>153.2</td>
<td>375.7</td>
<td>528.9</td>
</tr>
<tr>
<td>D72Z8R5S1</td>
<td>92.1%</td>
<td>15.5</td>
<td>170.6</td>
<td>367.9</td>
<td>538.5</td>
<td>97.3%</td>
<td>11.6</td>
<td>135.5</td>
<td>388.4</td>
<td>523.9</td>
</tr>
<tr>
<td>D72Z8R5S2</td>
<td>91.7%</td>
<td>16.2</td>
<td>178.3</td>
<td>366.1</td>
<td>544.4</td>
<td>96.2%</td>
<td>14.6</td>
<td>168.8</td>
<td>384.3</td>
<td>531.3</td>
</tr>
<tr>
<td>D72Z8R5S3</td>
<td>91.4%</td>
<td>17.5</td>
<td>191.8</td>
<td>365.0</td>
<td>556.8</td>
<td>96.9%</td>
<td>11.2</td>
<td>130.4</td>
<td>386.8</td>
<td>517.2</td>
</tr>
<tr>
<td>D72Z8R10S1</td>
<td>91.7%</td>
<td>17.8</td>
<td>195.9</td>
<td>366.2</td>
<td>562.1</td>
<td>96.6%</td>
<td>12.5</td>
<td>145.0</td>
<td>385.9</td>
<td>530.9</td>
</tr>
<tr>
<td>D72Z8R10S2</td>
<td>91.0%</td>
<td>18.8</td>
<td>205.3</td>
<td>363.2</td>
<td>568.5</td>
<td>96.3%</td>
<td>13.1</td>
<td>151.3</td>
<td>384.7</td>
<td>536.0</td>
</tr>
<tr>
<td>D72Z8R10S3</td>
<td>88.9%</td>
<td>20.9</td>
<td>222.8</td>
<td>355.2</td>
<td>578.0</td>
<td>96.0%</td>
<td>13.2</td>
<td>153.1</td>
<td>383.3</td>
<td>534.6</td>
</tr>
<tr>
<td>D72Z72R2S1</td>
<td>77.7%</td>
<td>29.4</td>
<td>273.8</td>
<td>310.3</td>
<td>584.1</td>
<td>91.6%</td>
<td>15.1</td>
<td>165.8</td>
<td>365.8</td>
<td>531.6</td>
</tr>
<tr>
<td>D72Z72R2S2</td>
<td>77.8%</td>
<td>29.3</td>
<td>273.3</td>
<td>310.8</td>
<td>584.1</td>
<td>91.7%</td>
<td>14.8</td>
<td>162.7</td>
<td>366.2</td>
<td>528.9</td>
</tr>
<tr>
<td>D72Z72R2S3</td>
<td>82.0%</td>
<td>26.0</td>
<td>255.8</td>
<td>327.7</td>
<td>583.5</td>
<td>93.3%</td>
<td>13.8</td>
<td>153.8</td>
<td>372.6</td>
<td>526.4</td>
</tr>
<tr>
<td>D80Z8R5S1</td>
<td>91.6%</td>
<td>14.8</td>
<td>180.2</td>
<td>366.0</td>
<td>546.2</td>
<td>96.1%</td>
<td>13.8</td>
<td>176.3</td>
<td>384.1</td>
<td>560.4</td>
</tr>
<tr>
<td>D80Z8R5S2</td>
<td>91.3%</td>
<td>15.5</td>
<td>188.2</td>
<td>364.8</td>
<td>553.0</td>
<td>95.9%</td>
<td>13.9</td>
<td>177.0</td>
<td>383.2</td>
<td>560.2</td>
</tr>
</tbody>
</table>
As expected, the proposed method outperforms the myopic baseline policy. In all instances, more service requests are accepted when the proposed method is applied. This statement is statistically significant (p-value $< 10^{-7}$) for each of the 27 instances based on a paired t-test for blocks of 60 days. Recall that our test is based in 6000 simulated days. As expected, the rural instances are the most challenging from the service provider's viewpoint, which is evident from the fact that a large portion of the requests in these instances could not be scheduled and served. In the rural instances, the advantage of the proposed method is the most prominent. Indeed, the proposed method increased the acceptance ratio from 79.2% to 92.3% on average for the rural instances. For the urban instances, the average acceptance ratio is increased from 91.7% to 96.7%.

Moreover, the fleet is utilized considerably more efficiently, as reflected by the travel time's decrease per accepted request. For example, the saving in the total travel time per request is 33% on average. The range of the saving is 5-49%, depending mainly on the spatial configuration. Consequently, under our policy, the technicians spend more time providing service and less time on the road, as reflected by the columns of the travel and service time per technician. The improvement is particularly considerable for the rural instances and instances with a larger number of requests.

Next, we verify that all the three arguments ($RTS$, $RTR$, and $TT$) of the opportunity cost function $f$ are essential, by applying our method while eliminating each one of the arguments, one at a time. These reduced models can be viewed as special cases of the full model, with $\alpha$, $\beta$, or $\gamma$ fixed as 0. Figure 4 presents the estimated acceptance ratio for these three modified models. The results of the full model are displayed as well. In section B of the electronic appendix, we report the three partial models' performance in detail.
The full model obtains significantly better average results than each of the partial models (p-value < 0.001 in paired t-tests that compare the full model with each of the partial ones). It appears that the marginal effect of the RTS measure on the performance of the model is relatively moderate, while the effect of the TT and RTR measures is considerably more notable, especially in rural settings.

5.3 The merits of a multiday service horizon

Recall that the proposed booking mechanism allows technician appointments scheduling in several future days with multiple cutoff times. In contrast, previous literature focused on a service horizon that consists of a single period and overlooked the interactions between consecutive working days. In this section, we examine the contribution of considering a multiday service horizon in terms of the acceptance ratio. To this end, we repeated our experiment with a service horizon of one day ($H^p = 1$), for all the instances. Figure 5 presents the acceptance ratio obtained with $H^p = 1$ and $H^p = 3$ for both the baseline myopic method and the proposed method. In section C of the electronic appendix, we present the detailed results of this experiment.
It is clear from Figure 5 that both the methods benefit from considering a multiday service horizon, which justifies the additional effort required by the proposed O-SSTBP model. In both the methods, the acceptance ratio with $H^P = 3$ is consistently higher than that with $H^P = 1$. In terms of the accepted requests, the gain from extending the service horizon from one to three days is 2.5–6.9% and 0.6–4.8% when using the proposed method and myopic policy, respectively. We believe that the difference can be attributed to the fact that the myopic policy is less suited to exploit the additional options available when the service horizon is longer.

6 Conclusions
In the past, field service personnel’s booking was usually performed through a sequential interaction between the customers and service representatives in call centers. The current practice is increasingly based on using online services (such as web or mobile apps), which are convenient to present the customers with all the offered alternatives at once. In such a case, the process of determining the alternatives to offer requires an algorithmic solution. In this study, we present an effective framework that addresses this challenge in relatively realistic settings. The service horizon spans several days with different cutoff times, and the price of the service is exogenous and independent of the service time slot. Previous studies focused on the single cutoff case and overlooked the interactions between consecutive days. The multiday and multi cutoff case requires replacing the objective function of maximizing the number of accepted requests in a single planning period to one that maximizes the steady-state acceptance rate.

The successful implementation of the proposed methods is highly dependent on a reliable forecast of the future demand and accurate enough modeling of the user choice model. Obtaining these forecasts from the data collected in booking systems is an interesting challenge beyond this paper’s scope. Note that for effective demand forecasting, the data regarding the offers presented to the customers should be collected, in addition to the actual choices of the customers. To this end, A-B testing may also be instrumental.
A closely related line of research considers booking of home attended deliveries. In this case, the booking system should also consider the capacity constraint of the vehicles. Introducing this constraint into the proposed method affects only the routing module and is straightforward.

The literature regarding the booking of field service personnel is still sparse, and we believe that the following extensions of our multiday model require further investigation: (1) Considering several classes of customers that differ in their time preferences, value to the service provider, required skills of the technicians, and expected service duration. Each such class may have a different optimal booking policy. (2) If the provider has no capacity to serve a request during the service horizon, it may be extended to make additional service slots available. An interesting direction for future research is introducing a decision regarding the length of the service horizon into the model and allowing its dynamic extension. (3) Resolving concurrency issues. When a large service provider performs the booking process, the system is likely to interact simultaneously with several customers. The slots offered to a particular customer may or may not be available to others while the interaction commences. The optimal booking policy may be affected by these concurrency issues.

Acknowledgment
This research was supported by a grant from the Israel Science Foundation (ISF) [Grant no. 1367/17].

The first author is partially supported by a scholarship from the Shlomo-Shmeltzer institute for smart transportation in Tel Aviv University.

References

