The online multiday technician booking problem

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

Abstract
We present a model for policy optimization of online booking of mobile personnel over a multiple days horizon, with a different cutoff for each day. The system negotiates with each customer, wishing to place a service request, with the goal of maximizing the expected number of accepted requests in the long term. The demand forecast and user choice model are inputs of the policy optimization problem. The negotiation with the customers is performed in a single step: The system offers a list of possible time slots during the booking 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 list of offered time slots following each service request is constructed by maximizing the expected net gain from the offer. The net gain is calculated based on an opportunity cost estimated using supervised machine learning techniques in a simulation framework. The proposed method is benchmarked using 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 teams of technicians and couriers that visit customers at their locations has attracted considerable interest. Current technology allows for greater automation of the service provision process. The first stage of this process involves a negotiation between the service provider and the customer regarding the time slot for beginning the service. The provider should take into account spatial and temporal considerations when offering the time slots. In this work, we refer to these negotiations as the booking process and study the mechanism by which the provider should make her offers where the goal is to ultimately maximize the number of service requests that can be performed while satisfying certain resource constraints. In particular, we envision an online process in which the provider side is fully automated.

Specifically, as described in Section 3, we define the online multiday technician booking problem, in which the provider is the leader in the negotiation process. That is, the provider offers several possible time slots to the customer, and the customer decides whether to accept one of the offers and place a service request or to abandon the system. 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 service providers, such as home delivery companies.
During the negotiation, the service provider should consider the following trade-off: by offering many possible time slots, the provider can increase the probability of the customer accepting one of them. However, if the accepted requests require the technicians to travel for a long time, their availability to serve future requests is reduced.

The multiday planning horizon is divided into time slots of typically 2–3 hours, for which requests are assigned. Once the provider commits to serve a customer at 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 component, which is unknown. We model this situation using a multinomial logit (MNL) discrete choice model.

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 learning the parameters of a Cobb–Douglas function with the following arguments: (1) the remaining idle time in the slot of the most suitable technician after the customer request is inserted in the slot; (2) the remaining idle time in the daily route of the most suitable technician after the customer request is inserted in the slot; (3) the additional travel time required to serve the request. We use a simulation to learn the parameters of this function. The net gain from an offered slot that is accepted by a customer is one minus its opportunity cost. The list 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.

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, which is based on a supervised machine learning technique. Section 5 presents the results of a numerical experiment that compares the performances of the optimized policy to a naive 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 a certain stochastic process and during a booking period. Each request is accepted or rejected online upon arrival following a short negotiation. The booking period ends at a certain cutoff time that precedes the execution period, 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.

One stream in the literature on online booking addresses the problem primarily as a dynamic vehicle routing problem. Campbell and Savelsbergh (2005) studied a booking problem in which the information related to the potential delivery requests is known at the beginning of the booking period. Each potential delivery request is associated with a location, revenue, capacity consumption, and set of possible time slots for the arrival at the customer location. The probability of each potential request to be materialized is known and represented by a function that decreases as the cutoff approaches. In other words, the arrival process of the requests is stochastic. A fleet of homogenous vehicles is available to serve the
accepted requests. The objective function is to maximize the total revenue net of the routing costs. The authors present several decision criteria related to the acceptance of newly arrived delivery requests. Only the delivery requests that do not cause late arrivals at the precommitted requests are considered. The first criterion, DYN, is myopic in nature. In this case, a system accepts a delivery request if the (tentative) net profit gained from serving it, i.e., its revenue net of the additional routing costs, is positive. The other criteria are anticipatory and consider the potential requests that have not occurred yet in the decision process. Requests with a higher probability to occur have a stronger effect on the decision. The numerical experiments demonstrated the merit of considering the future requests in the decision process.

Campbell and Savelsbergh (2006) studied an online booking problem in which each delivery request is associated with revenue and a location. Each customer can select each time slot with a certain probability, and it is assumed that this information is known to the provider. The service provider presents the customer with a set of possible time slots. The time slots that do not cause late arrivals at precommitted delivery requests are presented to the customer. In addition, each presented slot is associated with a nonnegative discount. Next, the customer chooses one of the proposed time slots. The authors modeled the probability of choosing each time slot as a linear decreasing function of its associated discount. Furthermore, the authors devised a policy that determines the discounts associated with the time slots presented with each newly arrived delivery request. The objective function is to maximize the total revenue net of the routing costs and discounts. The proposed methods are used to calculate the prices dynamically upon the arrival of each request, and do not consider possible future requests. The numerical experiments demonstrated the superiority of the proposed incentive policy over more straightforward baseline policies.

Ehmke and Campbell (2014) studied an online booking problem in which each delivery request is associated with two possible time slots from a predefined set. The customer presents these slots to the provider sequentially, and the provider selects the first feasible slot with respect to her previously committed requests. The objective function is to maximize the number of accepted requests. The authors presented several possible criteria for the acceptance of delivery requests and considered time dependent and/or stochastic travel times. The numerical experiment results indicated that the criteria that consider the travel time stochasticity yield routing plans that minimize the actual lateness at the customers’ locations when the travel times are time dependent and stochastic.

Köhler et al. (2020) studied an online booking problem in which each delivery request is associated with a location. Upon the arrival of a request, the service provider presents the customer with a set of possible time slots with two possible lengths, long and short. They note that offering long time slots increases the chances that future requests will be accepted but at the cost increasing the probability that the current customer will abandon. The objective is to maximize the number of accepted customers. The authors present four decision criteria related to the current utilization of the fleet and the location of the delivery request. Numerical experiments demonstrated the merits of each criterion over a baseline policy that offers short time windows only.

Another stream in the research incorporates elements from vehicle routing and revenue management in a framework of stochastic dynamic programming. Asdemir et al. (2009) studied an online booking problem in which the delivery requests from the same geographical area arrive, one by one, and dynamically. 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;
however, the routing aspects are not considered. In addition, the provider determines a price for each of the presented slots. The customer chooses one of the proposed time slots or abandons the process according to a multinomial logit (MNL) discrete choice model. The utility in this model is determined by the presented price and a particular characteristic of each slot. For example, the afternoon slot may be more desirable by the average customer compared to a midday slot. The authors devised a policy that determines the prices to maximize the total profit of the provider. The problem was formulated as a Markov decision process (MDP) that identifies the opportunity cost. In other words, the future profit obtained if the newly arrived request is not served net off 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.

Yang et al. (2016) studied a more general setting in which each delivery request is associated with revenue, location, and capacity consumption. A fleet of homogeneous vehicles is available to perform the deliveries. The provider presents the customer with a set of possible time slots that maintain capacity constraints and do not cause late arrivals at the precommitted requests. Similar to the approach of Asdemir et al. (2009), the provider determines a price for each of the offered slots, and the customer chooses a slot according to the MNL model. The price determination problem is formulated as a stochastic dynamic program. The prices are determined with the aim of maximizing the total expected profit of the provider, i.e., the revenue net of the routing cost. The price of each slot is calculated upon the arrival of each request, based on an approximation of its opportunity cost. Two approximation heuristics are implemented: (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 true marginal cost. The marginal cost is estimated by hypothetically inserting the current request into the final plans of previous days and observing the contribution of the request to the total costs. The numerical experiments demonstrated the applicability of estimating the MNL model from historical data. Moreover, the paper emphasizes 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 by implementing an offline approximated dynamic programming (ADP) framework that calculates 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. The numerical experiments demonstrated the merit of dynamic prices compared to standard benchmarks.

Mackert (2019) studied an e-grocer delivery booking problem where customers are offered a subset of delivery slots to choose from with the goal of maximizing the retailer profit net of the delivery cost. The decisions made by the customers are modeled by the generalized attraction model (GAM) proposed by Gallego et al. (2015). The author advocates the GAM instead of the more commonly used special case MNL since it allows circumventing an overestimation in demand by reflecting customers’ ‘dissatisfaction’ of increasingly narrow assortments of time windows. The approximation of the opportunity cost of each request is inspired by the method presented by Klein et al. (2018). Next, a mathematical program is employed to find an optimal assortment of time slots to offer. Mackert (2019) studies a model that is very similar to ours, where the main difference is the multiple cutoff property
that we consider. In terms of solution method, it uses a clever value approximation approach that is based on a solution of a deterministic mathematical program with nominal values of expected future demand.

Ulmer and Thomas (2019a) studied an online booking problem in which each delivery request is associated with a revenue, location and capacity consumption. Upon the arrival of a request, the service provider decides whether the request is accepted or rejected. The objective function is to maximize the sum of the revenues. The authors apply an ADP framework and devise a new value function approximation (VFA) procedure that combines parametric and non-parametric VFA methods. Numerical experiments demonstrated the superiority of the new procedure over standard ADP technics.

Finally, Ulmer and Thomas (2019b) studied an online booking problem in which a single vehicle operating the following day is available to serve the accepted requests. The vehicle departs from the depot at the beginning of the day and returns to the depot at the end of the day. A service request is accepted if serving it does not violate the deadline for the end of the vehicle’s working day. When a service request is accepted, an estimated arrival time is associated with it and communicated to the customer. In this work, the authors developed a method to estimate the arrival time at the service request. Specifically, the authors suggested that considering the possible future developments of a route as well as its current state is beneficial. According to the proposed scheme, the arrival time estimator for a newly arrived service request can be obtained as a function of three state variables: the total travel time of the route (not including the service times), travel time from the depot to the newly arrived request and the remaining slack time of the vehicle. This function is estimated based on many offline simulations (training runs) of a single booking period. A series of test runs were performed to demonstrate the superiority of this supervised machine learning (ML) method, in terms of the actual accuracy of the arrival time estimators, over a myopic estimation procedure that does not account for the arrival (and acceptance) of future service requests.

Table 1 summarizes the various attributes of the studies pertaining to online booking problems. Next, we discuss the literature pertaining to certain closely related topics, specifically, the tactical planning of online booking systems and dynamic vehicle routing. In particular, we consider the case in which requests continue to arrive after the cutoff time.

### Table 1: Comparison of existing literature pertaining to online booking problems

<table>
<thead>
<tr>
<th>Paper</th>
<th>Routing decisions</th>
<th>Negotiation protocol and choice model</th>
<th>Pricing</th>
<th>Multiperiod</th>
<th>Anticipatory</th>
<th>Solution method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Campbell and Savelsbergh (2005)</td>
<td>Yes</td>
<td>The customer presents a set of slots, and the provider either selects one of them or rejects the request</td>
<td>-</td>
<td>-</td>
<td>Yes</td>
<td>Estimate the net profit of each request using certain fixed criteria.</td>
</tr>
<tr>
<td>Campbell and</td>
<td>Yes</td>
<td>The provider knows the probability of the request</td>
<td>Yes</td>
<td>-</td>
<td>-</td>
<td>Incentives are offered to maximize the</td>
</tr>
<tr>
<td>Authors</td>
<td>Yes. Time dependency and stochasticity in travel times</td>
<td>Customer accepting each possible offer and the effect of the incentives that she may present.</td>
<td>-</td>
<td>-</td>
<td>Yes</td>
<td>MDP</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------</td>
<td>---</td>
<td>---</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>Ehmke and Campbell (2014)</td>
<td>Yes</td>
<td>The customer presents up to two slots sequentially, and the provider selects the first feasible slot</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Estimate feasibility and accept any feasible offer made by the customer</td>
</tr>
<tr>
<td>Köhler et al. (2019)</td>
<td>Yes</td>
<td>The provider presents a set of slots at various lengths to the customer. The customer either selects a slot or abandons the system</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Heuristic decision criterion for the length of each offered slot.</td>
</tr>
<tr>
<td>Asdemir et al. (2009)</td>
<td>-</td>
<td>The provider presents a set of slots and prices, and the customer either chooses a slot or abandons according to an MNL model</td>
<td>Yes</td>
<td>-</td>
<td>Yes</td>
<td>MDP</td>
</tr>
<tr>
<td>Yang et al. (2016)</td>
<td>Yes</td>
<td>Similar to Asdemir et al. (2009)</td>
<td>Yes</td>
<td>-</td>
<td>Yes</td>
<td>Approximate the opportunity cost by using historical data. The prices are optimized using a closed form solution.</td>
</tr>
<tr>
<td>Yang and Strauss (2017)</td>
<td>Approximated based on the number of accepted requests in each zone</td>
<td>Similar to Asdemir et al. (2009)</td>
<td>Yes</td>
<td>-</td>
<td>Yes</td>
<td>ADP</td>
</tr>
<tr>
<td>Ulmer and Thomas (2019a)</td>
<td>Yes</td>
<td>Customer places a delivery request and the provider</td>
<td>-</td>
<td>-</td>
<td>Yes</td>
<td>ADP</td>
</tr>
<tr>
<td>Authors</td>
<td>Year</td>
<td>Decision</td>
<td>Description</td>
<td>Forecast Method</td>
<td>Opportunity Cost Approximation</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
<td>----------</td>
<td>-------------</td>
<td>-----------------</td>
<td>-------------------------------</td>
<td></td>
</tr>
<tr>
<td>Ulmer and Thomas (2019b)</td>
<td></td>
<td>either accepts it or rejects it</td>
<td>Any feasible request is accepted, and the estimated arrival time is communicated to the customer</td>
<td>-</td>
<td>Yes</td>
<td>Forecast using ML</td>
</tr>
<tr>
<td>Mackert (2019)</td>
<td></td>
<td>Yes</td>
<td>The provider presents a set of slots, and the customer either chooses a slot or abandons according to a GAM</td>
<td>-</td>
<td>-</td>
<td>Yes</td>
</tr>
<tr>
<td>This paper</td>
<td></td>
<td>Yes</td>
<td>The provider presents a set of slots, and the customer either chooses a slot or abandons according to an arbitrary discrete model</td>
<td>-</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

*Tactical planning* is the process concerned with medium term decisions related to the operations of the service provider. Such decisions may include shift planning, dividing the service area into zones, and assigning personnel to these zones. Such planning is an offline process that is performed before the online booking. An effective tactical plan aims to minimize the cost of providing the service while allowing the routes created during the online booking process to maintain the desired service level and achieve other operational goals.

Agatz et al. (2011) studied the problem of allocating the delivery time slots to zones. To this end, a continuous approximation approach as well as an integer programming framework are applied. The generated allocations are referred to as *schedules*. Hernandez et al. (2017) suggested a method to select a schedule from a predefined set based on the cost of its associated tactical routing plan. The plans are devised by applying heuristics that model the related optimization problem as a periodic vehicle routing model. Bühler et al. (2016) claimed that an accurate prediction of the routing costs is crucial in the planning process. They construct this prediction by solving a related mixed integer programming (MIP) model.

Restrepo et al. (2019) devised a master plan for the operation of a delivery fleet. The plan defines the working hours for each courier, as well as the related hourly delivery zones and quotes. A scenario based two stage stochastic programming model for the problem is presented and solved by an L-shaped algorithm. Klein et al. (2019) demonstrated that the static differential pricing of time slots is beneficial for maximizing the total profit of the service provider. Furthermore, the authors presented a related MIP
formulation to determine the prices, in which the customers are assumed to follow a nonparametric rank based choice model.

The literature related to dynamic vehicle routing is quite extensive. For a comprehensive survey, the interested readers may refer to Pillac et al. (2013) and Psaraftis et al. (2016). In the following, we discuss several studies regarding dynamic routing that consider stochastic requests.

Bent and Van Hentenryck (2004) studied a dynamic vehicle routing problem with time windows (VRPTW) with the objective of maximizing the number of served customers. The authors presented a multiple scenario approach (MSA) to solve the problem. This approach employs stochastic information while maintaining multiple alternative routing plans. The merit of both components was demonstrated by a numerical experiment. Ghiani et al. (2009) studied a dynamic pickup and delivery problem (PDP) and showed that algorithms that anticipate the future requests tend to be superior to reactive myopic algorithms. Ghiani et al. (2012) studied a dynamic traveling salesperson problem (TSP) and presented two possible anticipatory solution approaches, namely, sample scenario planning and anticipatory insertion. The numerical experiments indicated that the latter approach may exhibit a comparable performance as that of the former approach, albeit with a significantly lower computational burden.

Soeffker et al. (2017) studied a dynamic vehicle routing problem and suggested that the superiority of anticipatory algorithms, in terms of the number of served customers, may be at the expense of reducing the service level at the periphery of the service zone. Ulmer (2019) formulates a similar problem as an MDP and presented two solution approaches for the problem, namely, a myopic re-optimization of the routing plan and an anticipatory ADP framework. Voccia et al. (2019) studied a dynamic PDP with the objective of maximizing the number of served requests, presented an MDP formulation for the problem, and implemented an adapted version of the MSA for the solution.

This paper is the first to study a multiday booking setting, 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 planning horizon is rolled forward by one day at the beginning of each working day to allow the handling of requests for a predefined booking horizon. In contrast to the single cutoff case, a multi cutoff framework requires optimization of the steady state performance of the booking system. An effective policy optimization method based on a supervised machine learning technique is introduced, and a numerical experiment that demonstrates its merits is performed.

3 Problem definition

The online multiday technician booking problem (O-MDTBP) is stated as follows: service requests arrive according to a certain stochastic process over an infinite horizon. Each request is characterized by its location, service time, and certain preferences regarding the starting time of the service. The infinite horizon is divided into working days, and each day is divided into several nonoverlapping service windows (e.g., 8–10, 10–12, 12–14, 14–16, 16–18). A combination of the service day and windows is referred to as a slot. Upon arrival, each request can be either rejected or scheduled to a slot in one of the next \( H^P \) working days, not including the current day. This period is referred to as the booking horizon. A cutoff time is defined before the beginning of each working day, \( h \), and requests that arrive after the cutoff can be scheduled in the new booking horizon that starts on day, \( h + 1 \).
The accepted requests are served by a fleet of \( M \) homogenous technicians available in each working day. The technicians depart from a depot, not before a predefined time \( a_0 \), and has to return to the depot no later than \( b_0 \). In other words, the working day duration is \( L = b_0 - a_0 \).

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

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

The set of displayed time slots 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 set of displayed slots 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 are determined after the cutoff time, and all the accepted requests must be served within their assigned time slots. This problem is a special case of the vehicle routing problem with time windows (VRP-TW), and it is not part of the scope and contribution of this study.

A solution for the O-MDTBP 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-MDTBP. In this paper, we use an MNL model; however, any discrete choice model could be used. The set of the time slots offered to a customer is denoted by \( S \). In addition, we denote the alternative of abandoning the system by 0. That is, the set of alternatives available to the customer is \([0] \cup S \). Each alternative is associated with a utility \( U_s = u_s + \varepsilon_s \). Here, \( u_s \) is the known expected value of the utility, and \( \varepsilon_s \) are independent and identically distributed according to the Gumble distribution. Next, the probability of selecting an alternative, \( s \), is

\[
p^s = \frac{e^{u_s}}{\sum_{i \in [0] \cup S} e^{u_i}}
\]

Moreover, we assume that all the customers have the same expected utility, \( u_s \), from each of the alternatives.

We divide the service area into \( Z \) zones. The requests from zone \( z \in \{1 \ldots Z\} \) arrive according to a Poisson process. Each arrival process is time dependent, relating to the hour of the day, i.e., the expected number of requests from zone \( z \) at hour \( t \) of the day is denoted by \( \lambda_{zt} \). We assume that the \( \lambda_{zt} \) values over the relevant horizon are known. Furthermore, we assume that the demand for the service within each zone and at each hour of the day is uniformly distributed. Note that this configuration is general enough to handle the various zoning schemes, ranging from a rough partitioning of large cities to quarters to points representing a single block or building. At the strategic planning step, it is essential to
define the zones such that the demand pattern within each zone is at least approximately uniform. The task of estimating and forecasting the demand patterns and defining the zones is crucial for a successful booking scheme; however, this task is beyond the scope of this study. We note that the partition of the service area to the zones is performed only to define the spatial distribution of the requests, and the zones do not impose any limitations on the routes of the technicians. In other words, if desirable, the same technician may serve customers from different zones in the same time slot.

Next, we assume that the service time, denoted by \( \tau \), is identical over all the requests. Finally, the travel time between any two locations \( j \) and \( k \) in the space, \( t_{jk} \), is known. The travel time may be based on metric distances (e.g., Euclidian) or, in the discrete case, can be given explicitly by a matrix (e.g., obtained from a GIS).

4 Methodology

Upon the arrival of a service request, the booking system has to construct a list of time slots that is offered to the customer to choose from. Recall that the customer may select one of these slots or abandon the system. At any particular moment, the booking system maintains a tentative set of routes and schedules for all the technicians and the entire planning horizon. Once a customer selects a time slot, his request is inserted into one of the tentative routes, and the schedule is updated.

The booking system constructs the list of offered time slots for each customer request with the goal of maximizing the expected number of requests that will be eventually served. The list is constructed by solving an optimization problem based on an estimation of the opportunity cost of each request. The opportunity cost is approximated by a Cobb–Douglas function that is learned using simulations.

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 machine learning framework for learning 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 customers that will either be rejected or will abandon the system as a result of the resources being allocated to the current request.

Consider a route \( r' \) that represents the set of customers 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 customers in \( r' \) as well as customer \( i \) is said to be feasible if the slots of all the previous customers in \( r' \) are not changed, and \( i \) starts the service at time slot \( s \).

We model the opportunity cost function as a Cobb–Douglas function that is calculated with respect to a given feasible route that includes a newly added request \( i \) at slot \( s \). Based on \((r, r', i, s)\), one can calculate the following three measures:
\[ RTS(r, s) \quad \text{Remaining idle time in slot } s \text{ (after the insertion of } i \text{ into route } r \text{ at slot } s) \]. The technician is considered idle when no service or travel are carried out.

\[ RTR(r, s) \quad \text{Remaining idle time (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 \).

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,

\[
f(r, r', s) = \begin{cases} 
RTS(r, s)^\alpha RTR(r, s)^\beta TT(r', r)^\gamma, & \text{if route } r \text{ is feasible} \\
0, & \text{otherwise}
\end{cases} \tag{1}
\]

where \( \alpha, \beta, \gamma \) are parameters that characterize the opportunity cost function \( f \) and may be either positive or negative. Note that \( f(r, r', s) > 1 \) implies that serving the current customer in route \( r \) at slot \( s \) incurs an expected loss of more than one future request, and thus, accepting this request is not worthwhile.

The measures \( RTS, RTR, \) and \( TT \) capture the state of the system in a concise manner and contain sufficient information to effectively estimate the opportunity cost. 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. In Section 4.4, we present a method for automating the learning of appropriate values of \( \alpha, \beta, \gamma \). It is worth mentioning that while the \( TT \) leads to decisions that are myopic in nature, the \( RTR \) and \( RTS \) combine the local and myopic considerations with anticipatory ones.

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 (1) by adding a small \( \epsilon > 0 \) to each of the measures. Specifically,

\[
f(r, r', s) = \begin{cases} 
(RTS(r, s) + \epsilon)^\alpha (RTR(r, s) + \epsilon)^\beta (TT(r', r) + \epsilon)^\gamma, & \text{if route } r \text{ is feasible} \\
0, & \text{otherwise}
\end{cases} \tag{1'}
\]

### 4.2 The routing and scheduling oracle

In this section, we present the procedure to determine, for a given technician and slot, the minimum insertion cost of a given request as well as the position in the route that achieves this cost. This estimation is later used to construct a list of offers to the customer, and once the customer selects a slot, to tentatively schedule and route this appointment. It should be noted that the routing and scheduling problem is not the focus of this study, and the proposed oracle can be replaced by any other fast procedure. A pseudocode of the routing and scheduling procedure is presented in Figure 1. This procedure is performed upon the arrival of a service request, denoted by \( i \).

**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

\[
\text{for } s \in S \\
C[s].f \leftarrow \infty \\
\text{for } k \in K \\
r' \leftarrow \text{current route of } k \text{ in the day of slot } s \\
\text{for } pl = 0 \text{ to } |r'| \\
r \leftarrow \text{route of } k \text{ after inserting request } i \text{ in position } pl \text{ to } r' \text{ at the earlist time in } s \\
\text{if route } r \text{ exists that is feasible for slot } s \\
\text{calculate } \text{RTS}(r, s), \text{RTR}(r, s), \text{TT}(r', r) \\
\text{if } f(r, r', s) < C[s].f \\
C[s].tech \leftarrow k \\
C[s].f \leftarrow f(r, r', i, s) \\
C[s].route \leftarrow r \\
S' = \{s \in S | C[s].f \leq 1\}
\]

Output: $S', C$

**Figure 1:** Pseudocode for finding a set of candidate slots

The list $S'$ stores the set of all the slots at the next $H^p$ days with an expected opportunity cost of up to $1$. Recall that more expensive slots should be ruled out for consideration for the offer made to the customer, since accepting these slots is expected to reduce the total number of serviced requests.

To construct $S'$, all the slots in the booking horizon $S$ are scanned and the opportunity cost of offering these slots is evaluated. To this end, 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, which includes 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 obtained by the Cobb–Douglas function 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. The technician’s route and schedule with the minimal opportunity cost are found for each slot. Next, a list with all the slots with an expected opportunity cost that is not greater than one, along with the best-found technicians and routes, is 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 $\text{RTS}, \text{RTR}, \text{and TT}$. 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, $TT = t_{ji} + t_{ik} - t_{jk}$.

The $\text{RTS}$ is calculated as follows. Let $u_j$ denote the starting times of service at the location of customer $j$, 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 \( s \), \( u_k = \max(a_s, u_j + \tau + t_{jk}) \). Next, 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 the list of offered slots

After calculating the approximated opportunity cost of scheduling the request at each slot, the planner can construct a list of slots to offer that will 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 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 offer, and an auxiliary variable \( P^s \) that holds the probability that the customer will select slot \( s \).

\[
\text{maximize } \sum_{s \in S'} P^s (1 - c[s,j]) \quad (2)
\]

subject to

\[
P^s = \frac{e^{u_s x_s}}{\sum_{q \in S'} e^{u_q x_q} + 1} \quad \forall s \in S' \quad (3)
\]

\[
x_s \in \{0,1\} \quad \forall s \in S' \quad (4)
\]

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 (3) relates the auxiliary variables \( P^s \) to the offered slots represented by \( x \). 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 \( e^0 = 1 \). 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 \( \sum_{s \in S'} P^s < 1 \), which represents the fact that in the MNL model, the probability of a customer abandonment is always positive.

In the numerical experiment, the optimization problem (2)–(4) is solved by enumerating all the subsets of \( S' \). A booking 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. For longer booking horizons, Mackert (2019) presented a linearized model for a generalization of (2)-(4) and noted some theoretical properties that allow solving it in polynomial time.

Once the list of optimal slots 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 \( C[s] \), as prescribed by the algorithm presented in Figure 1. The resulting plan is tentative, and the actual routes and schedule may be re-optimized later.

We note that while the discussion here and our numerical study in Section 5 are based on the MNL model, any discrete choice model that can assign selection probabilities to items in an offered set can be used.

4.4 Learning the parameters

In this section, we present an automated learning procedure for fitting values for the parameters of the opportunity cost estimation model. 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; however, convergence to a global optimum cannot be ensured. However, in Section 5, we provide numerical evidence for the effectiveness of the proposed heuristic.

The proposed learning procedure relies on a variant of the steepest descent method. The procedure has two building blocks: 1) estimation of the performance of a current solution; 2) ongoing creation of candidate improved solutions. To estimate the expected number of accepted requests for a given set of parameters, a discrete event simulation is performed. Specifically, we simulate a sufficient number of working days to obtain a good estimator. The candidate solutions are generated using an iterative search framework, in which a neighborhood of the current solution is created and evaluated in each iteration of the search. Next, the current solution is updated, and the process is repeated until a stopping criterion is met.

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. In other words, \( 3^3 = 27 \) solutions (including the current solution itself) comprise \( \mathcal{N} \). Let \( \Delta \) denote a nonnegative step size for the parameters in the current iteration of the search. Next, \( \mathcal{N} \) is obtained as the following cartesian product:

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

The performance of each solution in \( x \in \mathcal{N} \) is evaluated by simulation. If no improving solutions are found, \( x^{(n+1)} \leftarrow x^{(n)} \), 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.
However, if an improving solution exists in $\mathcal{N}$, let $x^{(n')} = (\alpha^{(n)} + \Delta n_\alpha, \beta^{(n)} + \Delta n_\beta, \gamma^{(n)} + \Delta n_\gamma)$ be the best solution in $\mathcal{N}$, where $n_\alpha, n_\beta, n_\gamma \in \{-1,0,1\}$. Next, we initiate a line search along a segment of the ray from $x^{(n)}$ at the direction of $x^{(n')}$, that is, in

$$\mathcal{M} = \{(\alpha, \beta, \gamma)| (\alpha^{(n)} + l\Delta n_\alpha, \beta^{(n)} + l\Delta n_\beta, \gamma^{(n)} + l\Delta n_\gamma)\}$$

where $l \in \mathbb{R}_+$. We use the golden section method to find a locally optimal solution $x^{(n+1)}$ in the line $\mathcal{M}$. 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 $x^{(n)}$.

The learning 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

This section describes the performed numerical experiments. 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 booking horizon with multiple cutoff times. Section 5.4, examines the effect of the scale of the system on the proposed booking mechanism.

5.1 Description of problem instances

In this section, we describe the settings of the benchmark instances used in the main experiment reported in Section 5.2. Some of the parameter values are diversified in Sections 5.3 and 5.4 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 nonoverlapping 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 booking horizon is three days, $H^p = 3$. Therefore, 15 potential time slots may be available for the service of a newly arrived request. The expected utilities of obtaining the service in each of the five slots on the next day, i.e., the first day of the booking 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>

Table 2: Expected utility of each possible slot

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.
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. It is natural that the arrival rate is considerably low during the nighttime.

![Figure 2: Fraction of requests arriving at each hour of the day](image)

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 $2/3$ of the total working time of the six technicians, i.e., 2,400 minutes. For example, for instances with 80 customers, $\tau = \frac{2400}{80} = 30$ minutes.

In all the problem instances, the locations of the customers 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, that is, $Z = 8$ (See Section 3). For the rural configuration, 72 points are generated to represent the centers of zones. In this case, $Z = 72$, and $x_z \sim U(2,98), y_z \sim U(2,98), z = 1 \ldots 72$.

Upon the arrival of each request in a given zone $z$, its location, represented by the Euclidian coordinates $(x,y)$, is generated as follows. 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. In all of our test instances, the arrival rates of the requests are $\lambda_{zt} = \frac{D \cdot d_t}{Z}$ for all $z = 1 \ldots Z$ and $t = 1, \ldots 24$. 

16
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 3: Example of the locations pertaining to the requests in each spatial configuration](image)

A training and testing streams of requests are generated for each instance. The training stream is used in the learning process applied to 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, \((\alpha, \beta, \gamma)\), obtained in the learning 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 estimated acceptance ratio represents the steady state. The length of the simulation and the warmup period were determined and verified by extensive statistical analysis.

For each unique combination of spatial configuration (urban, suburban, rural) and value of \(D\) (64, 72, 80 requests per day), we generated 3 instances with different random zone’s centers and random streams of requests. The total number of such benchmark instances is 27. A python script that reproduces this dataset and its detailed description is available in the electronic appendix that can be downloaded from [http://www.eng.tau.ac.il/~talraviv/Publications/](http://www.eng.tau.ac.il/~talraviv/Publications/).

The instances are namecoded according to the number of requests \(D\), number of zones \(Z\), 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 8 different settlements with a radius of 10 each (suburban setting).

**5.2 Performance evaluation of the proposed method**

In this section, we compare the performance of the proposed method 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 learning process begins with an initial solution: \(\alpha^0 = 0, \beta^0 = 0, \gamma^0 = 0\). The initial step size is set as \(\Delta = 0.1\), and it is updated by a factor of \(\omega=2\). The numerical tolerance of the Cobb–Douglas function is \(\epsilon = 0.01\). The process stops after 4 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 in all the instances.

To benchmark our method, we use a baseline myopic policy for generating a list of slots to offer that is common in practice. 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), the insertion of the request to the plan is based solely on minimizing the additional marginal (and tentative) routing cost. The performance of the myopic policy and proposed policy were compared based on the same geographies and testing streams.

Table 4 presents statistics related to the performance of the proposed method 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 working time net of waiting for a technician, which equals the sum of its travel and service time. The acceptance ratio is calculated based on the number of accepted requests throughout the simulated period (excluding the warmup period) and the total number of requests during this period.

<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>93.1%</td>
<td>20.8</td>
<td>206.3</td>
<td>372.5</td>
<td>579.0</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>92.5%</td>
<td>21.1</td>
<td>208.7</td>
<td>370.2</td>
<td>578.9</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>89.9%</td>
<td>23.0</td>
<td>220.5</td>
<td>359.9</td>
<td>580.4</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>91.0%</td>
<td>22.4</td>
<td>217.0</td>
<td>364.0</td>
<td>581.0</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>89.6%</td>
<td>23.3</td>
<td>222.8</td>
<td>358.4</td>
<td>581.2</td>
<td>96.4%</td>
<td>14.2</td>
<td>142.6</td>
<td>385.8</td>
<td>528.4</td>
</tr>
<tr>
<td>D64Z8R10S3</td>
<td>87.4%</td>
<td>24.8</td>
<td>231.6</td>
<td>349.9</td>
<td>581.5</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>77.8%</td>
<td>32.6</td>
<td>271.0</td>
<td>311.5</td>
<td>582.5</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>78.0%</td>
<td>32.5</td>
<td>270.4</td>
<td>312.1</td>
<td>582.5</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>81.9%</td>
<td>29.2</td>
<td>254.7</td>
<td>327.7</td>
<td>582.4</td>
<td>94.9%</td>
<td>15.3</td>
<td>153.2</td>
<td>375.7</td>
<td>528.9</td>
</tr>
<tr>
<td>D72Z8R5S1</td>
<td>91.4%</td>
<td>19.8</td>
<td>217.1</td>
<td>365.1</td>
<td>582.2</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>90.9%</td>
<td>20.1</td>
<td>218.5</td>
<td>363.1</td>
<td>581.6</td>
<td>96.2%</td>
<td>14.6</td>
<td>168.8</td>
<td>384.3</td>
<td>533.1</td>
</tr>
<tr>
<td>D72Z8R5S3</td>
<td>88.3%</td>
<td>21.7</td>
<td>230.0</td>
<td>352.8</td>
<td>582.8</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>89.0%</td>
<td>21.4</td>
<td>228.1</td>
<td>355.6</td>
<td>583.7</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>88.3%</td>
<td>21.8</td>
<td>231.0</td>
<td>352.6</td>
<td>583.6</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>85.7%</td>
<td>23.5</td>
<td>241.6</td>
<td>342.2</td>
<td>583.8</td>
<td>96.0%</td>
<td>13.2</td>
<td>151.3</td>
<td>383.3</td>
<td>534.6</td>
</tr>
<tr>
<td>D72Z72R2S1</td>
<td>75.2%</td>
<td>31.5</td>
<td>284.0</td>
<td>300.5</td>
<td>584.5</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>75.4%</td>
<td>31.4</td>
<td>283.3</td>
<td>301.2</td>
<td>584.5</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>79.2%</td>
<td>28.3</td>
<td>268.2</td>
<td>316.4</td>
<td>584.6</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>89.3%</td>
<td>19.2</td>
<td>228.2</td>
<td>356.8</td>
<td>585.0</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>88.9%</td>
<td>19.4</td>
<td>229.3</td>
<td>355.4</td>
<td>584.7</td>
<td>95.9%</td>
<td>13.9</td>
<td>177.0</td>
<td>383.2</td>
<td>560.2</td>
</tr>
<tr>
<td>D80Z8R5S3</td>
<td>86.0%</td>
<td>21.1</td>
<td>241.3</td>
<td>343.9</td>
<td>585.2</td>
<td>96.7%</td>
<td>10.9</td>
<td>140.6</td>
<td>386.4</td>
<td>527.0</td>
</tr>
<tr>
<td>D80Z8R10S1</td>
<td>86.6%</td>
<td>20.8</td>
<td>239.7</td>
<td>346.1</td>
<td>585.8</td>
<td>96.2%</td>
<td>11.5</td>
<td>148.0</td>
<td>384.6</td>
<td>532.6</td>
</tr>
<tr>
<td>D80Z8R10S2</td>
<td>85.8%</td>
<td>21.2</td>
<td>242.5</td>
<td>343.0</td>
<td>585.5</td>
<td>95.9%</td>
<td>12.5</td>
<td>159.9</td>
<td>383.2</td>
<td>543.1</td>
</tr>
<tr>
<td>D80Z8R10S3</td>
<td>83.3%</td>
<td>22.8</td>
<td>252.6</td>
<td>333.1</td>
<td>585.7</td>
<td>95.5%</td>
<td>12.4</td>
<td>158.0</td>
<td>381.6</td>
<td>539.6</td>
</tr>
<tr>
<td>D80Z72R2S1</td>
<td>72.8%</td>
<td>30.5</td>
<td>295.3</td>
<td>290.9</td>
<td>586.2</td>
<td>91.4%</td>
<td>14.6</td>
<td>177.8</td>
<td>365.1</td>
<td>542.9</td>
</tr>
<tr>
<td>D80Z72R2S2</td>
<td>72.8%</td>
<td>30.4</td>
<td>295.1</td>
<td>291.1</td>
<td>586.2</td>
<td>91.4%</td>
<td>14.7</td>
<td>178.9</td>
<td>365.2</td>
<td>544.1</td>
</tr>
<tr>
<td>D80Z72R2S3</td>
<td>76.6%</td>
<td>27.5</td>
<td>280.1</td>
<td>306.1</td>
<td>586.2</td>
<td>93.2%</td>
<td>13.5</td>
<td>167.8</td>
<td>372.6</td>
<td>540.4</td>
</tr>
</tbody>
</table>
The proposed method clearly outperforms the myopic baseline policy. In all the instances, more service requests are accepted when the proposed method is applied. Moreover, the fleet is utilized considerably more efficiently, as reflected by the increase in the service time per technician and the decrease in the travel time per accepted request. In addition, under our policy, the technicians spend more time providing service and less time on the road, as reflected by the columns pertaining to the travel and service time per technician. The improvement is particularly considerable for the rural instances and instances with a larger number of requests.

As expected, the rural instances are most challenging from the service provider’s viewpoint, which is evident from the fact that a large portion of the requests in these instances, even with \( D = 64 \), could not be scheduled and served.

Next, we verify that all the three arguments (\( RTS \), \( RTR \) and \( TT \) measures) 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. Table 5 presents the estimated acceptance ratio for these three modified models. The results of the full model are displayed as well. The best acceptance ratios are presented in boldface font.

**Table 5: Comparison to special cases**

<table>
<thead>
<tr>
<th>Instance</th>
<th>Full model</th>
<th>Model without ( RTS )</th>
<th>Model without ( RTR )</th>
<th>Model without ( TT )</th>
</tr>
</thead>
<tbody>
<tr>
<td>D64Z8R5S1</td>
<td>97.5%</td>
<td>97.2%</td>
<td>91.5%</td>
<td>92.4%</td>
</tr>
<tr>
<td>D64Z8R5S2</td>
<td>96.5%</td>
<td>97.0%</td>
<td>89.6%</td>
<td>91.1%</td>
</tr>
<tr>
<td>D64Z8R5S3</td>
<td>97.0%</td>
<td>96.7%</td>
<td>89.6%</td>
<td>90.3%</td>
</tr>
<tr>
<td>D64Z8R10S1</td>
<td>96.8%</td>
<td>96.3%</td>
<td>87.8%</td>
<td>90.7%</td>
</tr>
<tr>
<td>D64Z8R10S2</td>
<td>96.4%</td>
<td>95.9%</td>
<td>85.9%</td>
<td>88.9%</td>
</tr>
<tr>
<td>D64Z8R10S3</td>
<td>96.1%</td>
<td>95.5%</td>
<td>85.5%</td>
<td>88.2%</td>
</tr>
<tr>
<td>D64Z72R2S1</td>
<td>92.2%</td>
<td>90.6%</td>
<td>80.7%</td>
<td>80.4%</td>
</tr>
<tr>
<td>D64Z72R2S2</td>
<td>92.2%</td>
<td>90.5%</td>
<td>81.1%</td>
<td>80.0%</td>
</tr>
<tr>
<td>D64Z72R2S3</td>
<td>93.9%</td>
<td>92.4%</td>
<td>84.4%</td>
<td>84.1%</td>
</tr>
<tr>
<td>D72Z8R5S1</td>
<td>97.3%</td>
<td>97.0%</td>
<td>93.8%</td>
<td>92.1%</td>
</tr>
<tr>
<td>D72Z8R5S2</td>
<td>96.2%</td>
<td>96.8%</td>
<td>90.9%</td>
<td>90.8%</td>
</tr>
<tr>
<td>D72Z8R5S3</td>
<td>96.9%</td>
<td>96.5%</td>
<td>89.9%</td>
<td>89.7%</td>
</tr>
<tr>
<td>D72Z8R10S1</td>
<td>96.6%</td>
<td>96.0%</td>
<td>90.7%</td>
<td>90.2%</td>
</tr>
<tr>
<td>D72Z8R10S2</td>
<td>96.3%</td>
<td>95.7%</td>
<td>90.9%</td>
<td>89.4%</td>
</tr>
<tr>
<td>D72Z8R10S3</td>
<td>96.0%</td>
<td>95.1%</td>
<td>88.3%</td>
<td>87.3%</td>
</tr>
<tr>
<td>D72Z72R2S1</td>
<td>91.6%</td>
<td>89.9%</td>
<td>81.7%</td>
<td>78.8%</td>
</tr>
<tr>
<td>D72Z72R2S2</td>
<td>91.7%</td>
<td>90.0%</td>
<td>82.2%</td>
<td>78.4%</td>
</tr>
<tr>
<td>D72Z72R2S3</td>
<td>93.3%</td>
<td>91.9%</td>
<td>85.0%</td>
<td>82.3%</td>
</tr>
<tr>
<td>D80Z8R5S1</td>
<td>96.1%</td>
<td>96.7%</td>
<td>93.5%</td>
<td>91.3%</td>
</tr>
<tr>
<td>D80Z8R5S2</td>
<td>95.9%</td>
<td>96.7%</td>
<td>90.6%</td>
<td>91.3%</td>
</tr>
<tr>
<td>D80Z8R5S3</td>
<td>96.7%</td>
<td>96.3%</td>
<td>90.5%</td>
<td>88.9%</td>
</tr>
<tr>
<td>D80Z8R10S1</td>
<td>96.2%</td>
<td>95.5%</td>
<td>91.6%</td>
<td>89.3%</td>
</tr>
<tr>
<td>D80Z8R10S2</td>
<td>95.8%</td>
<td>95.1%</td>
<td>89.8%</td>
<td>88.6%</td>
</tr>
<tr>
<td>D80Z8R10S3</td>
<td>95.5%</td>
<td>94.3%</td>
<td>88.5%</td>
<td>86.7%</td>
</tr>
<tr>
<td>D80Z72R2S1</td>
<td>91.4%</td>
<td>89.2%</td>
<td>82.7%</td>
<td>77.2%</td>
</tr>
<tr>
<td>D80Z72R2S2</td>
<td>91.4%</td>
<td>89.0%</td>
<td>82.4%</td>
<td>76.8%</td>
</tr>
<tr>
<td>D80Z72R2S3</td>
<td>93.2%</td>
<td>91.1%</td>
<td>86.0%</td>
<td>80.4%</td>
</tr>
</tbody>
</table>
The full model obtains significantly better results than the corresponding special cases (p-value < 0.001 in paired t-tests that compare the full model with each of the reduced ones). The acceptance ratio of the four models are visualized in Figure 4. It appears that the marginal effect of the RTS measure on the performance of the model is quite moderate, while the effect of the TT measure is considerably more notable.

![Figure 4: Acceptance ratio of the full model and its special cases](image)

### 5.3 The merits of a multiday booking horizon

Recall that the proposed booking mechanism allows the scheduling of technician appointments in several future days with multiple cutoff times. In this section, we study the merits of a multiday booking horizon in terms of the acceptance ratio. To this end, we reinitiate our learning mechanism with a booking horizon of one day, \((H^p = 1)\), for all the instances. Table 6 presents the acceptance ratio obtained with \(H^p = 1\) and \(H^p = 3\) for both the baseline myopic method and the proposed method. These results are visualized in Figure 5.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Myopic, (H^p = 3)</th>
<th>Proposed method, (H^p = 3)</th>
<th>Myopic, (H^p = 1)</th>
<th>Proposed method, (H^p = 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D64Z8R5S1</td>
<td>93.1%</td>
<td>97.5%</td>
<td>89.1%</td>
<td>93.8%</td>
</tr>
<tr>
<td>D64Z8R5S2</td>
<td>92.5%</td>
<td>96.5%</td>
<td>88.3%</td>
<td>92.8%</td>
</tr>
<tr>
<td>D64Z8R5S3</td>
<td>89.9%</td>
<td>97.0%</td>
<td>86.5%</td>
<td>92.0%</td>
</tr>
<tr>
<td>D64Z8R10S1</td>
<td>91.0%</td>
<td>96.8%</td>
<td>87.7%</td>
<td>92.4%</td>
</tr>
<tr>
<td>D64Z8R10S2</td>
<td>89.6%</td>
<td>96.4%</td>
<td>86.4%</td>
<td>91.7%</td>
</tr>
<tr>
<td>D64Z8R10S3</td>
<td>87.4%</td>
<td>96.1%</td>
<td>84.8%</td>
<td>90.5%</td>
</tr>
<tr>
<td>D64Z72R2S1</td>
<td>77.8%</td>
<td>92.2%</td>
<td>76.9%</td>
<td>86.3%</td>
</tr>
</tbody>
</table>
It is clear from Table 6 and Figure 5 that both the methods benefit from prolonging the booking horizon, which justifies the additional effort required by the proposed O-MDTBP model. In both the methods, the acceptance ratio with $H^p = 3$ is consistently higher than that with $H^p = 1$. The gain, in terms of the accepted requests, from extending the booking horizon from one to three days, is 2.5–6.9% and 0–4% 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 that are made available when the booking horizon is longer.

We note that the proposed method performs better than the myopic policy even with a booking horizon of a single day. Even in this case, the proposed method consistently yields a higher acceptance ratio than that of the myopic method with a booking horizon of three days.
5.4 Economies of scale

In this section, we test the effect of the size of the system on the booking mechanisms and, in particular, on the performance of the proposed method. To this end, we created streamlined instances with one technician, and the request arrival rate is reduced by a factor of six. The requests were drawn from the same geographies of the corresponding original instances. These instances represent a firm that serves the same market in the same region in which the market is divided among several smaller firms. It is expected that, regardless of the booking mechanism, the acceptance ratio of the smaller firm is lower; however, we have no initial hypothesis regarding the interaction between the scale and the booking mechanism. The learning procedure is reinitiated for these instances, and the results are visualized as shown in Figure 6.

It is apparent from Figure 6 that both the myopic and proposed booking mechanisms exhibit economies of scale. When the numbers of requests and technicians were proportionally reduced, the acceptance ratios decreased on average by 11.8% and 12.4% under the myopic policy and the proposed mechanism, respectively. The effect of the scale on the acceptance ratio is significant under both mechanisms, however, the interaction is not statistically significant (p-value=0.37 in a two sided paired t-test). This implies, that we have no strong evident to support the claim that one of the booking methods exhibits larger economies of scale.

Finally, note that even for the smallest possible service providing firm (with one technician), the proposed method still outperforms the myopic policy by a large margin. We can conclude that the advantage of the proposed method is robust to the scale of the system.

Figure 5: Merits of using a longer booking horizon
6 Conclusions

In the past, the booking of field service personnel was usually performed through a sequential interaction between the customers and service representatives in call centers. The current practice is increasingly based on online services (such as web or mobile apps), using which it is 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, in which the booking horizon spans several days with different cutoff times, and the price of the service is exogenous and independent of the service time slot. Recent studies usually adopt a different approach that is based on endogenous pricing and focused on the single cutoff case. The multi cutoff case requires replacing the objective function of maximizing the 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 that is beyond the scope of this paper. 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 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 negotiation policy from the service provider's viewpoint. (2) In cases in which the provider has no capacity to serve a request during the booking period, this period may be extended to make additional service slots available. An interesting direction for future research is to introduce a decision regarding the length of the booking period into the model and allow its dynamic extension. (3) Resolving concurrency issues. When the booking process is performed by a large service provider, the system is likely to negotiate simultaneously with several customers. The slots offered to a customer may or may not be available to others until this negotiation ends. The booking policy should be amendable according to such constraints.

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


