Online Advance Admission Scheduling for Services with Customer Preferences

Xinshang Wang, Van-Anh Truong
Department of Industrial Engineering and Operations Research, Columbia University, New York, NY, USA, xw2230@columbia.edu, vatruong@ieor.columbia.edu

David Bank, MD, MBA
Department of Pediatrics, NYPH Morgan Stanley Children’s Hospital, Columbia University Medical Center, New York, NY, USA, deb40@columbia.edu

We study web and mobile applications that are used to schedule advance service, from medical appointments to restaurant reservations. We model them as online weighted bipartite matching problems with non-stationary arrivals. We propose new algorithms with performance guarantees for this class of problems. Specifically, we show that the expected performance of our algorithms is bounded below by \( \max(\frac{1}{2}, 1 - \sqrt{\frac{2}{\pi k}} + O(\frac{1}{k})) \) times that of an optimal offline algorithm, which knows all future information upfront, where \( k \) is the minimum capacity of a resource. This is the tightest known lower bound. Furthermore, we show that \( \frac{1}{2} \) is the best constant relative performance that can be achieved. This performance analysis holds for any Poisson arrival process. Our algorithms can also be applied to a number of related problems, including display-ad allocation problems and revenue-management problems for opaque products. We test the empirical performance of our algorithms against several well-known heuristics by using appointment-scheduling data from a major academic hospital system in New York City. The results show that the algorithms exhibit the best performance among all the tested policies. In particular, our algorithms are 21% more effective than the actual scheduling strategy used in the hospital system according to our performance metric.

1. Introduction

We study advance admission scheduling decisions in service systems. Advance admission scheduling decisions are those that determine specific times for customers’ arrival to a facility for service.
Advance admission scheduling is used in many service industries. Restaurants reserve tables for
customers who call in advance. Healthcare facilities reserve appointment slots for patients who
request them. Airlines reserve flight seats for those who purchase flight tickets. Advance admission
scheduling enables service providers to better match capacity with demand because they control
customers’ actual arrivals to service facilities.

We formulate and analyze a model that generally captures such admission scheduling systems.
For concreteness, we focus on the example of MyChart, a digital admission-scheduling application
developed by Epic System. Epic is an electronic medical records company that is managing the
records of millions of health care providers and more than half of the patient population in the U.S.
(Husain 2014). Epic deploys MyChart to perform online scheduling of appointments through inter-
net portals. The use of applications like MyChart is part of a general trend in healthcare towards
providing electronic access to service through web and mobile applications (TechnologyAdvice
2015).

When a patient schedules an appointment over a web portal, MyChart first asks the patient for
the type of visit desired, whether it is for a physical exam, a consultation, a flu shot, etc. Next,
it asks for the beginning and end of the range of preferred dates. It then shows a menu with a
check box for morning and afternoon session for each day in the preferred date range. Patients
can select one or more preferred sessions. Finally, MyChart either offers the patient one or more
appointments, or states that no appointment can be found. We can conceive of many variations
over this basic interface.

Consider the following model of advance admission scheduling that captures MyChart as an
example. Consider a continuous time, finite horizon. There are multiple service providers. Each
provider offers a number of service sessions over the horizon, some in regular hours and some in
overtime. We call a session associated with a single provider a resource. There are $n$ resources
available over the horizon. All the resources are known. Each resource $j$ can serve $C_j$ customers.
We call $C_j$ the capacity of resource $j$. Each resource $j$ must be booked by time $t_j$ or it perishes
at time $t_j$. There are $m$ customer types. Patients of type $i$, $i = 1, \ldots, m$, arrive according to some known non-homogeneous Poisson process and make reservations through any of the modes made available by the provider, web, phone, or mobile. A patient of type $i$ generates a benefit of $r_{ij}$ when served with a unit of resource $j$. We assume that the type of customers can be observed at the time that they arrive to make an appointment, through the pattern of preferences that they indicate and any data stored in the system on their profiles. We require that customers arriving at time $t$ have weight 0 for all resources $j$ that perish at time $t_j < t$. The number of customer types can be kept finite by discretizing the horizon but this number can be very large. We will discuss this point shortly. When a customer arrives, a unit of an available resource must be assigned to her, or she must be rejected. Each unit of a resource can be assigned to at most one customer. We allow no-shows and the practice of overbooking to compensate for the effect of no-shows. The objective of the problem is to allocate the resources to the customers to maximize the expected total benefit of the allocation.

Our advance-reservation model is essentially an online weighted bipartite matching problem. The resources in our model, when partitioned into units, can be seen as nodes on one side of a bipartite graph. All the customers correspond to nodes on the other side that are arriving online. The type of each arriving customer is determined by a time-varying distribution.

This resource-allocation model can be found in many other applications. We summarize three such applications below.

**Ad allocation.** In a typical display-ad allocation problem, e-commerce companies aim at tailoring display ads for each type of customers. Each ad, which corresponds to a unit of a resource, is often associated with a maximum number of times to be displayed. Knowing the arrival rates of future customers, the task is to make the most effective matching between ads and customers.

**Single-leg revenue management.** A special case of our model is the classic single-leg revenue-management problem in which all resources to be allocated are available at the same time. Customers who bring a higher benefit correspond to higher-fare classes. The decision is how to admit
or reject customers, given the time remaining until the flight and the current inventory of available seats.

**Management of opaque products.** Internet retailers such as Hotwire or Priceline often offer a buyer an under-specified or *opaque product*, such as a flight ticket, with certain details such as the exact flight timing or the name of the airline withheld until after purchase. We assume that demand for each opaque product is exogenous and independent of the availability of other products. When demand occurs, a decision is made to assign a specific product to that demand unit. Knowing the arrival rates of all demands, we want to maximize the total expected revenue by strategically assigning specific products.

Our contributions in this work are as follows:

- We provide the first general, high-fidelity model of advance admission scheduling that captures customer preferences across different resources. We allow non-stationary arrivals and no-shows. We model the advance admission-scheduling problem as an online weighted bipartite matching problem with non-stationary arrivals and propose new algorithms with guarantees on the relative performance.

- We prove the tightest known performance bound for the online matching problem with non-stationary stochastic arrivals. Specifically, we prove that a primitive algorithm, which we call the *Separation Algorithm*, has expected performance that is bounded by $\max(\frac{1}{2}, 1 - \sqrt{\frac{2}{\pi}} \frac{1}{k} + O(\frac{1}{k}))$ times that of an optimal offline algorithm, which knows all future information upfront, where $k$ is the minimum capacity of a resource. Furthermore, we show that $\frac{1}{2}$ is the best constant relative performance that can be achieved. Our performance bound improves upon the lower bound of Alaei, Hajiaghayi and Liaghat (2012). Moreover, it is close to an upper bound on the performance of the Separation Algorithm that the same authors found.

We obtain our bound by analyzing a novel *bounded Poisson process*. This is a Poisson process to which we apply a sequence of reflecting barriers. The process arises in the dual of an optimization problem that characterizes our performance bound. The behavior of this process is very
complex, with no known closed-form description. We managed to obtain a closed-form approximate characterization of the process.

- We improve on the Separation Algorithm by devising a novel bid-price-based algorithm, called the Marginal Allocation Algorithm, that is much more practical. First, the Allocation Algorithm is non-randomized, therefore more stable. Second, it is fair in the sense that it never rejects a high-priority customer but accepts a low-priority customer, assuming that their arrival times and preferences are the same. We prove that the Marginal Allocation Algorithm has the same theoretical performance guarantee as the Separation Algorithm. In addition, in numerical experiments, we show that it achieves much better practical performance.

- Our model also has application in other important problems such as display-ad allocation and opaque revenue management. For the display-ad allocation problem, we give the tightest known performance bound for an algorithm, assuming non-stationary arrivals and arbitrary mean demand. For the opaque revenue-management problem, we are the first to study online allocation policies for model with an arbitrary number of products and time-varying arrival rates.

- We test the empirical performance of our algorithm against several well-known heuristics by using appointment-scheduling data from a department within a major academic hospital system in New York City. The results show that our scheduling algorithms perform the best among all tested policies. In particular, our algorithm is 21% more effective than the actual scheduling strategy used in the hospital system according to our performance metric.

2. Literature Review

2.1. Appointment Scheduling

Our work is related to the literature on appointment scheduling. This area has been studied intensively in recent years (Guerriero and Guido 2011, May et al. 2011, Cardoen et al. 2010, Gupta 2007). A large part of this literature considers intra-day scheduling, in which the number of patients to be treated on each day is given or is exogenous, and the task is to determine an efficient sequence of start times for their appointments. Another part of the literature considers multi-day scheduling, in
which patients are dynamically allocated to appointment days. Some works in this literature focus on the number of patients to be served today, with the rest of the patients remaining on a waitlist until the next day. This paradigm is called allocation scheduling. See, for example, Huh, Liu, and Truong (2013), Min and Yih (2010), Ayvaz and Huh (2010), Gerchak, Gupta and Henig (1996). Recently, more works have focused on the problem of directly scheduling patients into future days. This paradigm is called advance scheduling. This paper considers an advance scheduling model with multiple patient classes. In the literature of advance scheduling, Truong (2014) first studies the analytical properties of a two-class advance scheduling model and gives efficient solutions to an optimal scheduling policy. For the multi-class model, no analytical result is known so far. Gocgun and Ghate (2012) and Patrick, Putterman, and Queyranne (2008) propose heuristics based on approximate dynamic programming for these problem, but have not characterized the worst-case performance of these heuristics. We propose the first online scheduling policy with performance guarantees for a very general multi-class advance-scheduling problem.

Our advance-scheduling model captures the preferences of patients in a general way. Patient preferences are an important consideration in most out-patient scheduling systems. In the literature considering patient preferences, Gupta and Wang (2008) considers a single-day scheduling model where each arriving patient picks a single slot with a particular physician, and the clinic accepts or rejects the request. Our model can be seen as a multi-period generalization of their work. We also characterize the theoretical performance in an online setting, whereas they use stochastic dynamic programming as the modeling framework and develop heuristics. Feldman et al. (2014) study how to offer sets of open appointment slots to a stream of arriving patients over a finite horizon of multiple days, given that patients have preferences for slots that can be captured by the multinomial logit model. Their work is strongly influenced by assortment-planning problems. An important observation, which was first made by Gupta and Wang (2008), is that there is a fundamental difference between many advance admission-scheduling problems and assortment-planning problems. In admission scheduling, we can often work with revealed preferences, whereas
in assortment-planning problems, decisions are made with knowledge only of a distribution of customer preferences. Working with revealed preferences allows for a more efficient allocation of service compared to working with opaque preferences. It also leads to more analytically tractable models.

### 2.2. Online Resource Allocation

Our work is closely related to works on online matching problems. Traditionally, the online bipartite matching problem studied by Karp et al. (1990) is known to have a best competitive ratio of 0.5 for deterministic algorithms and $1 - 1/e$ for randomized algorithms. For the online weighted bipartite matching problem that we consider, the worst-case competitive ratio cannot be bounded by any constant. Many subsequent works have tried to improve performance ratios under relaxed definitions of competitiveness.

Specifically, three types of assumptions are commonly used. The first type of assumption is that each demand node is independently and identically (i.i.d.) picked from a known set of nodes. Under this assumption, Jaillet and Lu (2014), Manshadi et al. (2012), Bahmani and Kapralov (2010), Feldman et al. (2009) propose online algorithms with competitive ratios higher than $1 - 1/e$ for the cardinality matching problem, in which the goal is to maximize the total number of matched pairs. Haeupler et al. (2011) study online algorithms with competitive ratios higher than $1 - 1/e$ for the weighted bipartite matching problem. Our definition of competitive ratio is the same as theirs. Our model is also similar, but we allow a more general arrival process of demand nodes in which the distribution of nodes can change over time. Previous analyses depend crucially on the fact that demand nodes are i.i.d. in order to simplify the expression for the probability that any demand node is matched to any resource node. The expression becomes much more complex, and the arguments break down in the case that demand arrivals are no longer i.i.d.

The second type of assumption is that the sequence of demand nodes is a random permutation of an unknown set of nodes. This random permutation assumption has been used in the secretary problem (Kleinberg 2005, Babaioff et al. 2008), adword problem (Goel and Mehta 2008) and the
bipartite matching problem (Mahdian and Yan 2011, Karande et al. 2011). Kesselheim et al. (2013) study the weighted bipartite matching problem with extension to combinatorial auctions. Our work is different from all of these in that the non-stationarity of arrivals in our model cannot be captured by the random permutation assumption.

The third type of assumption made is that each demand node requests a very small amount of resource. The combination of this assumption and the random-permutation assumption often leads to polynomial-time approximation schemes (PTAS) for problems such as adword (Devanur 2009), stochastic packing (Feldman et al. 2010), online linear programming (Agrawal et al. 2014), and packing problems (Molinaro and Ravi 2014). Typically, the PTAS proposed in these works use dual prices to make allocation decisions. Under this third assumption, Devanur et al. (2011) study a resource allocation problem in which the distribution of nodes is allowed to change over time, but still needs to follow a requirement that the distribution at any moment induce a small enough offline objective value. They then study the asymptotic performance of their algorithm. In our model, the amount capacity requested by each customer is not necessarily small relative to the total amount of capacity available. Therefore, the analysis in these previous works does not apply to our problem.

In our model, the arrival rates, or the distribution of demand nodes, are allowed to change over time. This non-stationarity poses new challenges, because it cannot be analyzed with existing methods. At the same time, it is an essential feature in our model because it allows us to capture the perishability of service capacity in the applications that we consider. When a resource perishes within the horizon, the demand for that resource drops to 0. Such a demand process must be time-varying. This important feature has received only limited attention so far. Ciocan and Farias (2012) consider an allocation model with a very general arrival process, but their allocation policy has performance guarantee only when the arrival rates are uniform. In this paper, we allow arrival processes to be non-homogeneous Poisson processes with arbitrary rates.

Our algorithms solves a linear program and uses its optimal solution to make matching decisions. The idea of using optimal solutions to a linear program is natural and has been used by several
previous works mentioned above. For example, Feldman et al. (2009), Manshadi et al. (2012), Haeupler et al. (2011), and Kesselheim et al. (2013) have used similar algorithms to obtain constant competitive ratios, albeit for different demand models.

While revising this paper, we discovered the paper of Alaei, Hajiaghayi and Liaghat (2012), which solves an online matching problem with non-stationary arrivals in a discrete-time setting. They propose an algorithm similar to our Separation Algorithm, which is a primitive algorithms that we analyze initially and later improve upon. They prove that this algorithm achieves a competitive ratio of at least $1 - \frac{1}{\sqrt{k}} - \frac{3}{\sqrt{2\pi k}}$ and at most approximately $1 - \frac{1}{\sqrt{2\pi k}}$, where $k$ is the minimum capacity of a resource. Compared to Alaei, Hajiaghayi and Liaghat (2012), we prove a stronger lower bound of $\max(1/2, 1 - \frac{2}{\sqrt{\pi k}} + O(\frac{1}{k}))$ on the competitive ratio for our Separation Algorithm, using a few of the same ideas but largely different techniques, as we will elaborate on in Section 5. Thus, our lower bound is more similar to their upper bound. We also point out that the Separation Algorithm is not practical because it might assign customers to resources that are already exhausted, while there are still other open resources. More importantly, because of randomization, it might reject a high-priority customers, but accept a low-priority customer at nearly the same time. For this reason, we propose a new “bid-pricing” algorithm, based on the Separation Algorithm, that avoids all of the above problems. We prove that the improved algorithm has the same theoretical performance guarantee, and has much better computational performance as tested on real data. Finally, we analyze the asymptotic performance of our algorithms, whereas they do not.

2.3. Revenue Management

Our work is also related to the revenue-management literature. We refer to Talluri and van Ryzin (2004) for a comprehensive review of this literature. Traditional works in this area assume that demands for products are exogenous and independent of the availability of other products (Lautenbacher and Stidham 1999, Lee and Hersh 1993, Littlewood 1972). The decision is whether to admit or reject a customer upon her arrival. Our model reduces to this admission-control problem in the special case that the resources are identical and are available at the same time.
When customers are open to purchase one among a set of different resources, our model controls which resource to assign to each customer. Thus, our model captures the problem of managing opaque products. Sellers of an opaque product conceal part of the products’ information from customers. Sellers have the ability to select which specific product to offer after the purchase of opaque product. This enables the seller to more flexibly manage their inventory. Opaque products are often sold at a discount compared to specific products, making them attractive to wider segments of the market. These products are common in internet advertising, tour operations, property management (Gallego et al. 2004) and e-retailing. Customers purchase an opaque product if the declared characteristics fit their preferences. The buyer agrees to accept any specific product that meets the opaque description. In our model, a specific product corresponds to a node on the right side of a bipartite graph. A unit of demand for an opaque product corresponds to a node on the left that connects to all of the specific products contained in the opaque product. The weight of an edge corresponds to the revenue earned by selling the opaque product.

Previous works related to opaque products include Gallego and Phillips (2004), Fay and Xie (2008), Petrick et al. (2010), Chen et al. (2010), Lee et al. (2012), Gönsch et al. (2014) and Fay and Xie (2015). Due to the problem of large state space, most analyses focus on models with very few product types. For systems with many product types, some pricing and allocation heuristics are known. There is numerical evidence that much of the benefit of opaque products can be obtained by having two or three alternatives (Elmachtoub and Wei 2013). However, when a retailer has a large number of alternative products, it is unclear how to design such an opaque product. Our work is the first to study online allocation policies with constant performance guarantees for the management of an opaque product with an arbitrary number of alternatives.

Our model assumes independent demands, i.e., the demand for each product is exogenous and independent of the availability of other products. Many recent works in revenue management consider endogenous demands, which means that customers who find their most preferred product unavailable might turn to other products. Examples of works on dependent demands include Gallego et al. (2004), Zhang and Cooper (2005), Liu and van Ryzin (2008) and Gallego et al. (2015).
One of the main characteristics of these models is that customer preferences cannot be observed until purchase decisions are made. In such situation, sellers only have a distributional information of customer preferences. This phenomenon does not apply to admission scheduling systems. In these systems, customer preference can be revealed before a unit of a resource is assigned. In MyChart, for example, the system is able to customize the appointment to offer to each patient after knowing the patient’s profile and availability. We assume that each customer’s preference is observed before a resource is assigned. Knowledge of preferences gives service providers the ability to improve the efficiency of the resource-allocation process by tailoring the service offered to each customer.

Our work is related to the still limited literature on designing policies for revenue management that are robust to the distribution of arrivals. Ball and Queyranne (2009) analyze online algorithms for the single-leg revenue-management problem. Their performance metric is the traditional competitive ratio that compares online algorithms with an optimal offline algorithm under the worst-case instance of demand arrivals. They prove that the competitive ratio cannot be bounded by any constant when there are arbitrarily many customer types. In our work, we relax the definition of competitive ratio, and show that our algorithms achieve a constant competitive ratio (under our definition) for any number of customer types and for a more general multi-resource model. Qin et al. (2015) study approximation algorithms for an admission-control problem for a single resource when customer arrival processes can be correlated over time. They use as the performance metric the ratio between the expected cost of their algorithm and that of an optimal online algorithm. Our performance metric is stronger than theirs as we compare our algorithms against an optimal offline algorithm, instead of the optimal online policy. Qin et al. (2015) prove a constant approximation ratio for the case of two customer types, and also for the case of multiple customer types with specific restrictions. They allow only one type of resource to be allocated. In our model, we assume arrivals are independent over time, but we allow for multiple customer types and multiple resources without additional assumptions.
3. Problem Formulation

3.1. Model

There are $n$ resources known to be available at specific instants over a continuous horizon $[0, 1]$. There are $m$ customer types. Customers of type $i$ randomly arrive over the horizon according to a known non-homogeneous Poisson process with rate $\lambda_i(t)$, for $t \in [0, 1]$. Let $\Lambda_i \equiv \int_0^1 \lambda_i(t) \, dt$ be the expected total number of arrivals of type-$i$ customers.

Each resource $j$ has a capacity of $C_j$ units. Each resource $j$ must be booked by time $t_j$ or it perishes at time $t_j$.

When a customer arrives, one unit of capacity of an available resource must be assigned to the customer, or the customer must be rejected. A customer of type $i$ earns a benefit $r_{ij}$ if assigned to resource $j$. The objective is to allocate the resources to the customers to maximize the expected total benefit from all of the allocated resources.

3.2. Definition of Competitive Ratios

Let $\delta_i$ be the actual total number of arrivals of type $i$ customers. We must have $E[\delta_i] = \Lambda_i$. An offline algorithm knows $\delta = (\delta_1, \delta_2, ..., \delta_m)$ at the beginning of the horizon. Let $OPT(\delta)$ be the optimal offline benefit given the number of arrivals $\delta$. Note that an optimal offline algorithm does not need to know the time of each arrival, as the algorithm essentially solves a maximum weighted matching problem, between the customers and resources. An online algorithm, however, does not know the entire sample path of future arrivals, but only knows the arrival rates $\lambda_i(t)$, $i = 1, 2, ..., m$.

In this paper, we define the competitive ratio as the ratio between the expected benefit of an online algorithm and the expected benefit of an optimal offline algorithm

**Definition 1.** An online algorithm is $c$-competitive if its total benefit $ALG$ satisfies

$$E[ALG] \geq cE[OPT(\delta)],$$

where the expectation is taken over the random vector $\delta$ of arrivals.
4. Online Resource Allocation Algorithms

Computing an optimal dynamic allocation policy for our problem by dynamic programming is intractable due to the curse of dimensionality. Let $c_j(t)$ indicate the amount of resource $j$ that remains at time $t$. Let $V(c(t), t)$ be the optimal expected total benefit to go starting with state $c(t) = (c_1(t), c_2(t), ..., c_n(t))$ and at time $t$.

In this section, we propose two online algorithms that approximate the total benefit $V(c(t), t)$ by a sum of single-variable functions

$$\sum_{j=1}^{n} f_j(t, c_j(t)),$$

where $f_j(t, c_j(t))$ is a benefit function that approximates the optimal benefit that can be obtained from resource $j$ from time $t$ to the end of the horizon.

The first algorithm, which we call the Separation Algorithm, separates and optimizes the decisions for each single resource. We show that the Separation Algorithm is $0.5$-competitive using a simple and innovative Lagrangian duality approach. We will further show that if $k$ is the minimum capacity of any resource, then the competitive ratio of the Separation Algorithm can be improved to $\max(\frac{1}{2}, 1 - \sqrt{\frac{2}{\pi k}} + O(\frac{1}{k}))$. Finally, we prove that $0.5$ is the best possible constant competitive ratio for our model.

The second algorithm, which we call the Marginal Allocation Algorithm, improves on the Separation Algorithm by converting it to a bid-price algorithm, which can be easily applied to a real admission scheduling system.

Before presenting the two online algorithms, we first characterize an optimal offline algorithm and an upper bound on the optimal offline benefit.

4.1. Offline Algorithm and Its Upper Bound

In the offline case, the total number of arrivals $\delta_i$ of each customer type $i$ is known, and the exact arrival time is irrelevant. Given the $\delta_i$’s, the maximum offline benefit $OPT(\delta)$ is given by a
maximum weighted matching problem, which can be formulated as the following LP:

\[
\text{OPT}(\delta) = \max \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} r_{ij}
\]

s.t. \( \sum_{j=1}^{n} x_{ij} \leq \delta_i, \) for \( i = 1, 2, \ldots, m \)

\( \sum_{i=1}^{m} x_{ij} \leq C_j, \) for \( j = 1, 2, \ldots, n \)

\( x_{ij} \geq 0, \) for \( i = 1, 2, \ldots, m; \ j = 1, 2, \ldots, n. \) \hspace{1cm} (1)

where the decision \( x_{ij} \) is the number of type-\( i \) customers who are assigned to resource \( j \). Let \( \bar{x}(\delta) \) be an optimal solution to this LP. Then \( \text{OPT}(\delta) = \sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij} \bar{x}_{ij}(\delta) \).

We are interested in finding an upper bound on the expected optimal offline benefit \( \mathbb{E}[\text{OPT}(\delta)] \).

We next show that LP (2), which uses \( \mathbb{E}[\delta] \) instead of \( \delta \) as the total demand, gives such an upper bound:

\[
\max \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} r_{ij}
\]

s.t. \( \sum_{j=1}^{n} x_{ij} \leq \Lambda_i, \) for \( i = 1, 2, \ldots, m \)

\( \sum_{i=1}^{m} x_{ij} \leq C_j, \) for \( j = 1, 2, \ldots, n \)

\( x_{ij} \geq 0. \) \hspace{1cm} (2)

**Theorem 1.** The optimal objective value of (2) is an upper bound on \( \mathbb{E}[\text{OPT}(\delta)] \).

**Proof.** Since \( \sum_{j=1}^{n} \bar{x}_{ij}(\delta) \leq \delta_i \) and \( \sum_{i=1}^{m} \bar{x}_{ij}(\delta) \leq C_j \), we must have \( \sum_{j=1}^{n} \mathbb{E}[\bar{x}_{ij}(\delta)] \leq \mathbb{E}[\delta_i] = \Lambda_i \) and \( \sum_{i=1}^{m} \mathbb{E}[\bar{x}_{ij}(\delta)] \leq C_j \). Thus, \( \mathbb{E}[\bar{x}_{ij}(\delta)] \) is a feasible solution to the LP (2). It follows that the optimal objective value of (2) is an upper bound on

\[
\sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij} \mathbb{E}[\bar{x}_{ij}(\delta)] = \mathbb{E}[\text{OPT}(\delta)].
\]

Similar techniques have been used in revenue management to prove similar results (Gallego and van Ryzin 1997). \( \square \)
4.2. Separation Algorithm and Constant Competitive Ratio

The Separation Algorithm works by solving the LP (2) once, routing the customers to the resources according to an optimal solution to the LP (2). Then, for each resource separately, the algorithm optimally controls the admission of customers who have been routed to that resource. Using the LP information with respect to the expected number of arrivals (or sometimes, an estimate of the expected number of arrivals) is natural and has been used in several previous results (for example, Feldman et al. (2009), Manshadi et al. (2012), Haeupler et al. (2011), and Kesselheim et al. (2013)).

For the rest of this section, we assume without loss of generality that the capacity of each resource is 1. Our bound and algorithm will be independent of the capacity. In the next section, we shall show that the algorithm and bound can be improved as the minimum capacity increases beyond 1.

Let $\mathbf{x}^*$ be an optimal solution to the linear program (2). Whenever a customer of type $i$ arrives, the Separation Algorithm randomly and independently picks a candidate resource $j \in \{1, 2, ..., n\}$ with probability $x_{ij}^*/\Lambda_j$, regardless of the availability of resources. We say that this customer is routed to resource $j$. Then based on a further decision, the algorithm may either assign resource $j$ to the customer or reject the customer.

According to the Poisson thinning property, the arrival process of type-$i$ customers who will be routed to resource $j$ is a non-homogeneous Poisson process with rate

$$\lambda_{ij}(t) = \lambda_i(t)x_{ij}^*/\Lambda_i, \quad \text{for } 0 \leq t \leq 1.$$  

Viewing the random routing process as exogenous, each resource $j$ receives an independent arrival process with split rate $\lambda_{ij}(t)$ from each customer type $i$. Then for each resource $j$, the Separation Algorithm optimally controls the admission of customers who are routed to resource $j$. That is, when a type-$i$ customer is routed to resource $j$ at time $t$, the algorithm compares the benefit $r_{ij}$ of this customer with the optimal expected future benefit $f_j(t)$ that can be earned from customers who will be routed to resource $j$ after time $t$. (Note that once resource $j$ becomes unavailable, any
customer routed to resource \( j \) will be rejected.) The algorithm assigns resource \( j \) to the customer if \( r_{ij} \) is greater than \( f_j(t) \). Given the split rates \( \lambda_{ij}(t)'s \), the optimal expected future benefit \( f_j(t) \) can be computed by solving the well-known Hamilton-Jacobi-Bellman equation

\[
f_j'(t) = -\sum_{i=1}^{m} \lambda_i(t)x^*_{ij}/\Lambda_i \cdot (r_{ij} - f_j(t))^+
\]

with boundary condition \( f_j(1) = 0 \).

We call \( f_j(t) \) the benefit function of resource \( j \). Although the HJB equation is in continuous time, in practice, it can be computed by discretizing the horizon into periods. Furthermore, according to properties of the HJB equation, \( f_j(t) \) is decreasing in \( t \), which captures the fact that resources are expiring over time.

Below are the detailed steps of the Separation Algorithm.

1. Solve for an optimal solution \( x^* \) to the linear program (2).
2. For each resource \( j \in \{1, 2, ..., n\} \), compute the benefit function \( f_j(t) \) according to (4).
3. Upon an arrival of a type-\( i \) customer at time \( t \), randomly pick a number \( j \in \{1, 2, ..., n\} \) with probability \( x^*_{ij}/\Lambda_j \). Assign resource \( j \) to the customer if resource \( j \) is still available and \( r_{ij} \geq f_j(t) \).

The following lemma gives the total expected benefit of the Separation Algorithm.

**Lemma 1.** The expected total benefit of the Separation Algorithm is \( \sum_{j=1}^{n} f_j(0) \) at time 0. If resource \( j \) is available at time \( t \), the Separation Algorithm earns benefit \( f_j(t) \) from resource \( j \) in time \( [t, 1] \) in expectation.

**Proof.** According to the HJB equation (4), the Separation Algorithm earns \( f_j(0) \) from resource \( j \) in expectation. Therefore, the total expected benefit of the Separation Algorithm is \( \sum_{j=1}^{n} f_j(0) \).

\( \square \)

The Separation Algorithm has the appeal that, at any time and any given state, we can easily compute the total expected benefit of remaining resources by summing up the values of benefit functions of all available resources. More importantly, the marginal benefit of having an additional resource is exactly equal to the value of the benefit function. As a result of the convenience of
computing marginal benefit values, we can significantly improve the empirical performance of the Separation Algorithm by converting it into a bid-price algorithm, which we will discuss in the next section.

The following theorem states that the Separation Algorithm has constant performance guarantee.

**Theorem 2.** The Separation Algorithm is 0.5-competitive.

To prove this theorem, we first analyze the competitive ratio for a single-resource benefit-maximization problem. Specifically, we want to study the following performance ratio for resource $j$.

$$f_j(0) / \sum_{i=1}^{m} r_{ij} x^*_ij,$$  \hspace{1cm} (5)

where $\sum_{i=1}^{m} r_{ij} x^*_ij$ can be seen as an upper bound on the expected optimal offline benefit for resource $j$ (see Theorem 1), and $f_j(0)$ is the expected benefit of the Separation Algorithm for resource $j$.

In order to determine a lower bound of (5), we first normalize $\sum_{i=1}^{m} r_{ij} x^*_ij$ to 1, which is helpful because all benefit values can be scaled by an arbitrary constant without affecting performance ratios. We must then search for a lower bound on $f_j(0)$ by examining all possible combinations of problem data, including the arrival rates (3), benefit values $r_{ij}$, and all possible optimal solutions $x^*$ to the LP (2), subject to the normalization condition $\sum_{i=1}^{m} r_{ij} x^*_ij = 1$.

Since both $x^*$ and the arrival rates $\lambda_i(t)$ can be expressed in terms of the split rates

$$\lambda_i(t) = \sum_{j=1}^{n} \lambda_{ij}(t), \forall i = 1, 2, ..., m,$$

$$x^*_ij = \int_{0}^{1} \lambda_{ij}(t) dt, \forall i = 1, 2, ..., m, j = 1, 2, ..., n,$$

we will instead examine all values of $r_{ij}$'s and $\lambda_{ij}(t)$'s. This problem can be formulated as follows

$$\inf_{\lambda_{ij}(t), r_{ij}, i=1,...,m} f_j(0)$$ \hspace{1cm} (6)

s.t.  \hspace{1cm} $f_j'(t) = - \sum_{i=1}^{m} \lambda_{ij}(t) \cdot (r_{ij} - f_j(t))^+$ \hspace{1cm} (7)

$$\int_{0}^{1} \sum_{i=1}^{m} \lambda_{ij}(t) dt \leq 1$$ \hspace{1cm} (8)
\[ \int_0^1 \sum_{i=1}^m \lambda_{ij}(t) r_{ij} dt = 1 \quad (9) \]

\[ \lambda_{ij}(t), r_{ij} \geq 0, \quad i = 1, 2, \ldots, m \quad (10) \]

\[ f(1) = 0, \quad (11) \]

In this optimization problem, the decisions are \( \lambda_{ij}(t) \)’s and \( r_{ij} \)’s. Constraint (7) is the dynamic programming equation (4). Constraint (8) is equivalent to the capacity constraint in LP (2), namely

\[ \sum_{i=1}^m x_{ij}^* \leq 1, \]

which requires that the average number of customers who are routed to resource \( j \) is at most 1 (recall that the capacity of each resource is 1 and each customer requires a unit of resource according to our model). Constraint (9) is the normalization condition \( \sum_{i=1}^m r_{ij} x_{ij}^* = 1 \).

**Theorem 3.** The optimal objective value of problem (6) is at least 0.5.

Our proof shows that a Lagrangian relaxation of the problem has objective value that reduces to at least 0.5. These techniques differ from those of Alaei et al. (2012), who analyze the dual problem to obtain a bound.

**5. Capacity-Dependent Competitive Ratio**

When a resource has greater than unit capacity, the algorithm presented in the previous section treats each unit of the resource as a separate resource, and does not exploit the fact that these units are interchangeable. In this section, we show that we can improve the performance bound of the Separation Algorithm when resources have greater than unit capacities.

Recall that \( C_j \) denotes the capacity of resource \( j \), for \( j = 1, 2, \ldots, n \). We assume that all capacity values are positive integers. Let \( x^* \) be an optimal solution to (2). For each resource \( j \), the following is a dynamic program that optimally controls the admissions of the fraction \( x_{ij}^*/\Lambda_i \) of type \( i \) customers, for \( i = 1, 2, \ldots, m \), who are routed to resource \( j \).

\[ f_j'(t, c) = -\sum_{i=1}^m \lambda_i(t) x_{ij}^*/\Lambda_i \cdot (r_{ij} - f_j(t, c) + f_j(t, c - 1))^+, \quad (12) \]
where $f_j(t,c)$ is the expected total future benefit that can be earned from resource $j$ starting at time $t$ when there are $c$ units of resource $j$ still available. The boundary conditions are $f_j(1,c) = 0$ and $f_j(t,0) = 0$.

Let $c_j(t)$ be the remaining capacity of resource $j$ at time $t$. When a customer of type $i$ arrives at time $t$ and is routed to resource $j$, the (generalized) Separation Algorithm compares $r_{ij}$ with $f_j(t,c_j(t)) - f_j(t,c_j(t)-1)$. It offers resource $j$ if $c_j(t) \geq 1$ and

$$r_{ij} \geq f_j(t,c_j(t)) - f_j(t,c_j(t)-1),$$

and rejects the customer otherwise.

Since the dynamic program (12) optimally integrates the decisions for all the $C_j$ units of resource $j$, the value of the new benefit function $f_j(t,c)$ must be at least the sum of benefit functions that the original Separation Algorithm uses for each available unit of the resource. Then by a similar argument as the proof of Theorem 2, we can easily check that the total expected benefit of the generalized Separation Algorithm must be at least $\sum_{j=1}^{n} f(0,C_j)$. Therefore the algorithm is still 0.5-competitive.

More importantly, we expect that this generalized Separation Algorithm will have better performance when the capacity values are large, due to the integrated decisions made for each entire resource. To prove an improved bound for the case of general capacities, we focus on a single resource $j$. We will prove that the Separation Algorithm achieves a better competitive ratio for this resource. We will suppress the index $j$ of the resource in the rest of the section except when needed to avoid confusion.

Assume that the capacity of the fixed resource is $k$. Let $V_l(t) = f_j(t,k-l)$ be the optimal expected future benefit at time $t$ when the remaining capacity is $k-l$ (this notation will be more convenient for analysis), for $t \in [0,1]$ and $l = 0,1,\ldots,k-1$. The HJB equation defining $V_l(\cdot)$ is

$$\frac{dV_l(t)}{dt} = -\sum_{i=1}^{m} \lambda_i(t) x^*_i/\Lambda_i \cdot (r_{ij} - V_l(t) + V_{l+1}(t))^+ = -\sum_{i=1}^{m} \lambda_{ij}(t) \cdot (r_{ij} - V_l(t) + V_{l+1}(t))^+$$

with boundary conditions $V_k(t) = 0$ and $V_l(1) = 0$. 


We are interested in the performance ratio

\[ \frac{f_j(0,k)}{\sum_{i=1}^m x_{ij}^* r_{ij}} = \frac{V_0(0)}{\int_0^1 \sum_i r_{ij} \lambda_i(t) dt}, \]

where \( \sum_{i=1}^m x_{ij}^* r_{ij} = \int_0^1 \sum_i r_{ij} \lambda_i(t) dt \) is an upper bound on the optimal expected offline benefit. We want to find the smallest such ratio by examining all possible inputs \( r \) and \( \lambda(\cdot) \).

Note that at any time \( t \), the performance ratio can be lowered by replacing the problem instance with one in which there is only one type of customer arrival with rate \( \lambda(t) = \sum_{i=1}^m \lambda_i(t) \) and reward \( r(t) = \frac{\sum_{i=1}^m r_{ij} \lambda_i(t)}{\sum_{i=1}^m \lambda_i(t)} \), so that the worst-case instance has one customer type, and time-dependent reward function \( r(\cdot) \). This observation has also been made by Alaei, Hajiaghayi and Liaghat (2012).

Thus, to characterize the worst-case performance ratio, we only need to bound the ratio

\[ \frac{V_0(0)}{\int_0^1 r(t) \lambda(t) dt}, \]

for \( V \) defined as

\[ \frac{dV_i(t)}{dt} = -\lambda(t)(r(t) - V_i(t) + V_{i+1}(t))^+, \]

over all reward functions \( r(\cdot) \) and arrival rate functions \( \lambda(\cdot) \) such that the second constraint of (2) is satisfied, i.e.,

\[ \int_0^1 \lambda(t) dt \leq k. \]

Note that the first constraint of (2) is implicitly satisfied after we set

\[ \lambda(t) = \sum_{i=1}^m \lambda_i(t) = \sum_{i=1}^m \lambda_i(t)x_{ij}^*/\Lambda_i. \]

5.1. Homogenizing time

Without loss of generality, we can change the horizon length, the arrival process \( \lambda(\cdot) \) and the reward process \( r(\cdot) \) as follows, while keeping the ratio (13) unchanged:

1. If \( \int_0^1 \lambda(t) dt < k \), we can extend the horizon to length \( T > 1 \) by adding more arrivals with benefit 0. Thus, we can equivalently assume \( \int_0^T \lambda(t) dt = k. \)
2. Define a (virtual) time variable as
\[ \tilde{t} = \tilde{t}(t) \equiv \int_0^t \lambda(s)ds. \]
Note that \[ \tilde{t} \in [0, k]. \] Using this new time variable, we can define new benefit functions as
\[ \bar{V}_i(s) = V_i(\tilde{t}^{-1}(s)), \]
where we interpret \[ \tilde{t}^{-1}(s) \] as the first time \[ t \] that satisfies \[ \tilde{t}(t) = s. \] Similarly, we can define \[ \bar{r}(s) = r(\tilde{t}^{-1}(s)). \] Then we can equivalently transform the HJB equation for \[ V_i(t) \] as follows
\[ \frac{dV_i(t)}{dt} = \frac{d\bar{V}_i(\tilde{t})}{d\tilde{t}} \frac{d\tilde{t}}{dt} = \frac{d\bar{V}_i(\tilde{t})}{d\tilde{t}} \lambda(t) \]
\[ \Rightarrow \frac{d\bar{V}_i(\tilde{t})}{d\tilde{t}} \lambda(t) = -\lambda(t)(r(t) + V_{i+1}(t) - V_i(t))^+ = -\lambda(t)(\bar{r}(\tilde{t}) + \bar{V}_{i+1}(\tilde{t}) - \bar{V}_i(\tilde{t}))^+ \]
\[ \Rightarrow \frac{d\bar{V}_i(\tilde{t})}{d\tilde{t}} = -(\bar{r}(\tilde{t}) + \bar{V}_{i+1}(\tilde{t}) - \bar{V}_i(\tilde{t}))^+, \forall \tilde{t} \in [0, k] \]

This equation can be viewed as another HJB equation with arrival rate 1 and revenue function \[ \bar{r}(\cdot), \]
with boundary conditions \[ V_k(t) = 0 \] for \[ t \in [0, k] \] and \[ V_i(k) = 0 \] for \[ i = 0, 1, \ldots, k-1. \] Furthermore, the upper bound on the expected offline benefit can be transformed as
\[ \int_0^T r(t)\lambda(t)dt = \int_0^k \bar{r}(\tilde{t})d\tilde{t}. \]
In summary, we can equivalently transform the problem into one whose arrival rate is uniformly 1 and whose time horizon is \[ [0, k]. \]

5.2. Bound-revealing optimization problem

After applying the above transformations, we can write an optimization problem that reveals the competitive ratio as follows

\[
\begin{aligned}
\min_{r(t), V_i(t), i=0,1,\ldots,k-1; t\in[0,k]} & \quad V_0(0) \\
\text{s.t.} & \quad \frac{dV_i(t)}{dt} = -(r(t) + V_{i+1}(t) - V_i(t))^+, \forall i = 0, 1, \ldots, k-1; t \in [0, k] \\
& \quad \int_0^k r(t)dt = 1 \\
& \quad V_i(t) \geq 0, \forall i = 0, 1, \ldots, k-1; t \in [0, k] \\
& \quad r(t) \geq 0.
\end{aligned}
\]
Here the second constraint $\int_0^k r(t)dt = 1$ normalizes the upper bound on the expected offline benefit. By using $g_i(t) = -dV_i(t)/dt$ and replacing $(\cdot)^+$ with linear constraints, we can write the above problem equivalently as (note that $g_k(t) = 0, \forall t \in [0,k]$)

$$\begin{align*}
\min_{r(t), g_i(t), i=0,1,...,k-1; t\in[0,k]} & \int_0^k g_0(s)ds \\
\text{s.t.} & \quad g_i(t) \geq r(t) + \int_t^k g_{i+1}(s)ds - \int_t^k g_i(s)ds, \ \forall i = 0, 1, ..., k-1; \ \forall t \in [0,k] \\
& \quad \int_0^k r(t)dt = 1 \\
& \quad g_i(t) \geq 0, \ \forall i = 0, 1, ..., k-1; \ t \in [0,k] \\
& \quad r(t) \geq 0.
\end{align*}$$

Let $\alpha_i(t)$ be a dual variable for the first constraint, for all $i = 0, 1, ..., k-1$ and $t \in [0,k]$. Let $\beta$ be a dual variable for the second constraint. The dual problem is

$$\begin{align*}
\max_{\alpha_i(t), \beta} & \beta \\
\text{s.t.} & \quad \alpha_0(t) + \int_0^t \alpha_0(s)ds \leq 1, \ \forall t \in [0,k] \\
& \quad \alpha_i(t) + \int_0^t \alpha_i(s)ds \leq \int_0^t \alpha_{i-1}(s)ds, \ \forall i = 1, 2, ..., k-1; \ \forall t \in [0,k] \\
& \quad \beta \leq \sum_{i=0}^{k-1} \alpha_i(t), \ \forall t \in [0,k] \\
& \quad \alpha_i(t) \geq 0.
\end{align*}$$

This dual problem tries to maximize the minimum value of $\sum_{i=0}^{k-1} \alpha_i(t)$ with respect to $t$. The optimal $\beta$ is a lower bound on the competitive ratio that we seek to characterize.

**5.3. A dual-feasible solution for the bound-revealing problem**

We first show that a feasible solution to the dual problem (15) can be constructed based on a modification of a Poisson process. As we shall explain shortly, this is a Poisson process to which we apply a control, using a sequence of *bounding barriers*. We will use the solution obtained via this derived process to obtain a lower bound on the optimal value of the bound-revealing
optimization problem (14). We will refer to the process as a bounded Poisson process. Alaei, Hajiaghayi and Liaghat (2012) also prove their bound by working with a dual-feasible solution. However, we construct our dual-feasible solution differently using a novel method. Because our bound has to be tighter, our analysis of this solution is also much more involved.

Let \( t_0, t_1, t_2, \ldots, t_k \) be a sequence of time points such that \( 0 = t_0 < t_1 < \cdots < t_{k-1} < t_k = k \).

Let \( \{N(t)\}_{t \geq 0} \) be a (counting) Poisson process with rate 1. We apply an upper barrier to \( N(t) \) to obtain a new bounded process \( \{R(t)\}_{t \geq 0} \). Starting with an initial value 0 at time \( t_0 = 0 \), the barrier increases by 1 at times \( t_1, t_2, \ldots, t_{k-1} \). At these time points, the new bounded process has values

\[
R(t_i) = \max(i - 1, R(t_{i-1}) + N(t_i) - N(t_{i-1})), \forall i = 1, 2, \ldots, k - 1,
\]

with \( R(t_0) = R(0) = 0 \). And for \( t \in [t_i, t_{i+1}] \), we have

\[
R(t) = \max(i, R(t_i) + N(t) - N(t_i)), \forall i = 1, 2, \ldots, k - 1.
\]

Eventually,

\[
R(t_k) = R(k) = \max(k - 1, R(t_{k-1}) + N(k) - N(t_{k-1})).
\]

**Theorem 4.** There exists a feasible dual solution \( \beta^*, \alpha^*_i(t) \) for \( t \in [0, k] \), \( i = 0, 1, 2, \ldots, k - 1 \), such that

\[
\alpha^*_i(t) = P(R(t) = i) \beta^*, \forall t \in [0, k], i = 0, 1, \ldots, k - 1,
\]

\[
k(1 - \beta^*) = \beta^* \left[ k - \sum_{i=0}^{k-1} iP(R(k) = i) \right]
\]

for the bounded Poisson process \( R(t) \) as constructed above.

Given that \( \beta^* \) and \( \alpha^* \) are dual-feasible, we will next attempt to bound objective \( \beta^* \) by analyzing the process \( R(\cdot) \).

First we show that the times at which the barriers are applied are bounded by \( 1, 2, \ldots, k - 1 \).

**Theorem 5.** The time points \( t_1, t_2, \ldots, t_{k-1} \) constructed in the proof of Theorem 4 satisfy \( t_i \leq i \), for \( i = 1, 2, \ldots, k - 1 \).
Before proving Theorem 5, we first prove Lemmas 2 to 6, which characterize further the behavior of the process $R(t)$. These lemmas collectively show that when the barriers are applied at regular points starting at some time of the horizon, i.e., $t_i = i \; \forall i \geq l$ for some integer $l$, the time spent at the barriers must be monotone decreasing in the index $i$ for all $i \geq l$.

For ease of notation, let
\[
I_i \equiv \int_{t_i}^{t_{i+1}} 1(R(s) = i)ds
\]
be the total time that the bounded process $R(t)$ stays at the barrier $i$ during the interval $[t_i, t_{i+1}]$, for $i = 0, 1, \ldots, k - 1$. Note that $E[I_i] = \int_{t_i}^{t_{i+1}} P(R(s) = i)ds$. Let $P_i(\lambda)$ be the probability that a Poisson random variable with mean $\lambda$ is equal to $i$. Let $P_{\geq i}(\lambda)$ and $P_{\leq i}(\lambda)$ denote $\sum_{j=i}^{\infty} P_j(\lambda)$ and $\sum_{j=0}^{i} P_j(\lambda)$, respectively.

First, assuming that the barriers are applied at regular points $0, 1, \ldots, k - 1$, we can quantify the difference in expected time spent at each barrier, given different starting points for the process $R(\cdot)$.

**Lemma 2.** Given any $l \in \{1, 2, \ldots, k - 1\}$, if $t_l = l$ and $t_{l+1} = l + 1$, we must have
\[
E[I_i|R(l) = l - j] - E[I_i|R(l) = l - j - 1] = P_{\geq j+1}(1)
\]
for all $j = 0, 1, \ldots, l - 1$.

Next, assuming that the barriers are applied at regular points $0, 1, \ldots, k - 1$, we can bound differences in time spent at each barrier for successive pairs of starting points.

**Lemma 3.** Given any $l \in \{2, 3, \ldots, k - 1\}$, if $t_i = i$ for all $i = l, l+1, \ldots, k - 1$, we must have
\[
E[I_i|R(l) = l] - E[I_i|R(l) = l - 1] \geq e^{-1}(E[I_i|R(l) = l - 1] - E[I_i|R(l) = l - 2])
\]
for all $i = l, l+1, \ldots, k - 1$.

Using the previous result, we relax the assumption that all the barriers are applied at regular points $0, 1, \ldots, k - 1$. We assume now that the barriers are applied at regular times beyond a point. Under this condition, we show that the differences in time spent at successive barriers are increasing with the starting point of the process.
Lemma 4. Given any \( l \in \{1, 2, \ldots, k-2\} \), if \( t_i \leq i \) for \( i = 1, 2, \ldots, l \), and \( t_i = i \) for \( i = l+1, l+2, \ldots, k-1 \), we must have

\[
E[I_i|R(l) = l] - E[I_{i+1}|R(l) = l] \geq E[I_i|R(l) = l-1] - E[I_{i+1}|R(l) = l-1]
\]

for all \( i = l, l+1, \ldots, k-2 \).

Next, assuming that the barriers are applied at regular points \( 0, 1, \ldots, k-1 \), we show that the time spent by the process at each barrier is decreasing with the index of the barrier.

Lemma 5. If \( t_i = i \) for all \( i = 1, 2, \ldots, k-1 \), we must have \( E[I_i] \geq E[I_{i+1}] \) for all \( i = 1, 2, \ldots, k-2 \).

Proof. It is obvious that for any \( i \geq 1 \),

\[
E[I_i|R(1) = 1] \geq E[I_i|R(1) = 0],
\]

because when the starting position becomes lower, it is harder for the random process \( R(t) \) to reach the barrier at any later time. Since \( E[I_i|R(1) = 0] = E[I_i] \), and by symmetry, \( E[I_i|R(1) = 1] = E[I_{i-1}] \), we have \( E[I_{i-1}] \geq E[I_i] \) for all \( i \geq 1 \). \( \square \)

Finally, we relax the requirement of Lemma 5. We require only that the barriers be applied at regular points only after some time. We show that the time spent at the barriers are still decreasing.

Lemma 6. Given any \( l \in \{1, 2, \ldots, k-2\} \), if \( t_i \leq i \) for \( i = 1, 2, \ldots, l \), and \( t_i = i \) for \( i = l+1, l+2, \ldots, k-1 \), we must have

\[
E[I_i] \geq E[I_{i+1}]
\]

for all \( i = l, l+1, \ldots, k-2 \).

The idea of the proof of Theorem 5 is as follows. We will start by setting the barriers at times \( 0, 1, \ldots, k-1 \). We then successively reduce the values \( t_i, i = 0, 1, \ldots \), until the time spent at each barrier is no more than \( 1/\beta - 1 \). By the monotonicity shown in Lemma 6, this procedure must stop with the time spent at each barrier bounded above by \( 1/\beta - 1 \). If we change the value of \( \beta \), the time points \( t_1, t_2, \ldots, t_{k-1} \) that result from the above procedure must change continuously in \( \beta \). We simply choose \( \beta \) such that, when the procedure ends, the time spent at the last barrier is \( 1/\beta - 1 \), which implies that the time spent at all barriers is exactly \( 1/\beta - 1 \).
5.4. Computing the bound

First, we prove an inequality, which will be useful in computing our bound.

**Lemma 7.** For any \(x, y \in \mathbb{Z}\) and \(\lambda \in [0, k]\) such that \(x \geq y \geq k - 1 - \lambda\), we must have for any \(l = 0, 1, ..., k - 1\),

\[
\sum_{i=-l}^{l} P_{k-1+i-x}(\lambda) \leq \sum_{i=-l}^{l} P_{k-1+i-y}(\lambda).
\]

Finally, we derive our bound. The bound is simply a reduction of the equation

\[
k(1 - \beta^*) = \beta^*[k - \sum_{i=0}^{k-1} iP(R(k) = i)],
\]

which follows from Theorem 4. \(\beta^*\) is strictly greater than 0.5 for \(k \geq 2\). For example, when \(k = 2\), \(\beta^*\) satisfies

\[
3\beta + \beta e^{1/\beta - 3} = 2,
\]

from which we can obtain \(\beta^* \approx 0.615\).

**Theorem 6.**

\[
\beta^* \geq \frac{1}{1 + \frac{1}{k} \left[ \sum_{i=2k-1}^{\infty} iP_i(k) + 2 \sum_{i=1}^{k-1} iP_{i+k-1}(k) \right]}.
\]

**Corollary 1.** Assuming that the minimum capacity for each resource is \(k\), the competitive ratio for the Separation Algorithm is at least

\[
\beta^* \geq \frac{1}{1 + 2 \left( \frac{P_{>k}(k)}{k^2} + \frac{e^{-k/k}}{k^3} \right)} = 1 - \sqrt{\frac{2}{\pi}} \frac{1}{\sqrt{k}} + O\left(\frac{1}{k}\right).
\]

6. Marginal Allocation Algorithm

The Separation Algorithm, when carried out in practice, has several problems. First, it might route customers to unavailable resources when they can be better matched to other resources. Second, because of the random routing, it might unfairly accept a lower-priority customer after rejecting a higher-priority customer. In this section, we present the Marginal Allocation Algorithm which resolves these issues by converting the Separation Algorithm into a bid-price algorithm. We will
prove that the Marginal Allocation Algorithm has theoretical performance no worse than that of the Separation Algorithm.

The Marginal Allocation Algorithm uses the marginal benefit \( f_j(t, c_j(t)) - f_j(t, c_j(t) - 1) \) as a bid price for resource \( j \). When a customer of type \( i \) arrives, the Marginal Allocation Algorithm rejects the customer if \( r_{ij} < f_j(t, c_j(t)) - f_j(t, c_j(t) - 1) \) for all available resource \( j \); otherwise, it assigns this customer to resource

\[
\arg\max_j \{ r_{ij} - f_j(t, c_j(t)) + f_j(t, c_j(t) - 1) | j \text{ is available at time } t \}.
\]

To carry out this algorithm, we only need to compute the \( n \) benefit functions at the beginning of the horizon, thus reducing the space requirement to polynomial size. At any time \( t \), we only need to know the \( n \) benefit functions \( f_j(t, c_j(t)) \), for \( j = 1, 2, ..., n \), so as to make a decision.

The following theorem states that the Marginal Allocation Algorithm performs at least as well as the Separation Algorithm:

**Theorem 7.** The expected total benefit of the Marginal Allocation Algorithm is no less than that of the Separation Algorithm.

### 6.1. Asymptotic performance

We can show that the Marginal-Allocation Algorithm is asymptotically optimal as the system size tends to infinity. Talluri and van Ryzin (1998) are the first to study asymptotic behavior of bid-price control in network revenue management problems. Our proof follows theirs and subsequent proofs of similar results. Let \( C_j = \theta C_j \), for \( j = 1, 2, ..., n \), \( \lambda_i(t) = \theta \lambda_i(t) \), for \( i = 1, 2, ..., m \), where \( \theta \) is a system scaling parameter and the barred quantities are fixed. Let \( \bar{x}^* \) be an optimal solution for the system \((\bar{C}, \bar{\lambda}(t))\), then \( x^* = \theta \bar{x}^* \) is an optimal solution for the system \((C, \lambda(t))\). The following theorem guarantees that the performance of the algorithm approaches the optimal objective value of the LP (2) when \( \theta \) goes to infinity.

**Theorem 8.**

\[
\lim_{\theta \to \infty} \frac{\sum_{j=1}^n f_j(0, C_j)}{\theta r^t x^*} = \lim_{\theta \to \infty} \frac{\sum_{j=1}^n f_j(0, \theta C_j)}{\theta r^t \bar{x}^*} = 1,
\]

where \( x^* \) is an optimal solution to (2).
6.2. Upper bound on the competitive ratio

In the above analysis we have shown that 0.5 is a lower bound on the best competitive ratio. Next, we show that 0.5 is also an upper bound on competitive ratio of any online algorithm. That is, our algorithms achieve the best constant competitive ratio.

**Theorem 9.** The competitive ratio of any online algorithm is at most 0.5.

**Proof.** Consider a situation in which a single resource is available to be allocated. There are two types of customers who want to be matched to that resource.

- Type-1 customers arrive in time $[0, 0.5]$. Their arrival rate is very large in the period $[0, 0.5]$. In particular, $\Lambda_1 = \int_0^{0.5} \lambda_1(t)dt \gg 1$, so that we can ignore the event that no type-1 customer arrives. Their benefit for the resource is $r_1 = 1$.

- Type-2 customers arrive in time $[0.5, 1]$. They have a very small arrival rate. In particular, $\Lambda_2 = \int_{0.5}^{1} \lambda_2(t)dt \ll 1$. Their benefit for that resource is $r_2 = 1/\Lambda_2 \gg 1$.

Since $r_2 \gg r_1$, the offline algorithm will allocate the resource to a type-2 customer, if there is one. The probability that at least one type-2 customer arrives is $1 - e^{-\Lambda_2} = \Lambda_2 + o(\Lambda_2^2)$. With probability $1 - o(\Lambda_2)$, no type-2 customer will arrive, in which case the optimal offline algorithm will assign a type-1 customer (there are plenty of type-1 customers) and earn benefit $r_1 = 1$. In sum, the expected total offline benefit is

$$
  r_2(\Lambda_2 + o(\Lambda_2^2)) + r_1(1 - o(\Lambda_2))
  = 1/\Lambda_2(\Lambda_2 + o(\Lambda_2^2)) + 1 \cdot (1 - o(\Lambda_2))
  = 1 + o(\Lambda_2) + 1 - o(\Lambda_2)
  = 2 + o(\Lambda_2).
$$

The decision of an online algorithm is whether to allocate the resource to a type-1 customer during the first half of the horizon. If it does allocate the resource to a type-1 customer, the online algorithm earns benefit $r_1 = 1$. Otherwise, with probability $\Lambda_2 + o(\Lambda_2^2)$ it earns $r_2$, which equals
1 + o(Λ_2) in expectation. In sum, the expected benefit obtained by an online algorithm cannot exceed 1 + o(Λ_2). Thus, an upper bound of the competitive ratio is

\[
\frac{(1 + o(Λ_2))}{(2 + o(Λ_2))},
\]

which tends to 0.5 in the limit as Λ_2 → 0. □

6.3. Overbooking

Another issue that is common to all advance admission-scheduling systems is the issue of no-shows. When customers book in advance, events may transpire between the date of the booking and the planned date of service that cause customers to miss their appointments. Due to the frequent occurrence of no-shows, overbooking is commonly used in service industries. Suppose each customer has a no-show probability of \( p_j \) when assigned to resource \( j \), and incurs a cost of \( D_j \) when being denied getting resource \( j \). Then we can model the overbooking strategy by expanding capacities at additional costs. Assume that the no-show events are exogenous to both online and offline algorithm. For resource \( j \), the \( k \)th overbooked unit of capacity incurs an expected marginal cost of

\[
o_j(k) = D_j \cdot (1 - p_j) \cdot \left[ \sum_{l=0}^{k-1} \binom{C_j + k - 1}{l} p_j^l (1 - p_j)^{C_j + k - 1 - l} \right],
\]

where the value in the brackets represents the probability that, among the \( C_j + k - 1 \) customers who have already booked resource \( j \), at most \( k - 1 \) of them do not show up. The additional \( 1 - p_j \) in the product represents the probability that the \( k \)th overbooked customer does show up. Note that the marginal cost \( o_j(k) \) is independent of customer type, and is increasing in \( k \).

Assuming that the benefit \( r_{ij} \) is earned whether a customer of type \( i \) actually takes resource \( j \), the marginal benefit of allocating the \( k \)th overbooked unit of resource \( j \) to a type \( i \) customer is

\[
\tilde{r}_{i,j,k} = r_{ij} - o_j(k).
\]

When using this benefit value \( \tilde{r}_{i,j,k} \), we are treating each overbooked unit of resource \( j \) as a virtual slot to be allocated. Then, the theoretical bound of our algorithms still applies, with \( \tilde{r}_{i,j,k} \) being the benefit of expanded units.
Since \( r_{i,j,k} \leq r_{i,j} \) and \( r_{i,j,k} \) decreases in \( k \), an optimal offline algorithm, when allocating resource \( j \), will first fill in the \( C_j \) units of regular capacity and then assign customers to those virtual slots with lower values of \( k \). It will not use virtual slots with non-positive marginal benefit. Then, when \( b \) overbooked units of resource \( j \) are used under the optimal offline algorithm by the end, the total cumulative cost
\[
\sum_{k=1}^{b} o_j(k)
\]
(18)
is just the actual expected overbooking cost for resource \( j \).

7. Computing Algorithms

In some applications, the number of customer types \( m \) can be extremely large such that the size of LP (2) is too large to be dealt with in practice. For example, in the display-ad allocation problem, customers can have hundreds of different attributes (Ciocan and Farias 2014) and thus the dimension of customer type space can be huge. In such cases, it is hard to compute the benefit functions \( f_j(t) \) by directly solving the LP (2), due to the huge number of constraints.

In this section, we propose an alternative method that estimates the benefit functions \( f_j(t) \) by simulation, using only a subset of all customer types. The simulation algorithm works if the average number of arrivals \( \Lambda_i \) is very small for every customer type \( i \). (If \( \Lambda_i \) is large for certain type \( i \), one can randomly split the customers into multiple types, such that the arrival rate of each type is smaller.) The algorithm requires the ability to

- Randomly select a set \( S \) of customer types. Each of the \( m \) customer types has the same probability to be selected into \( S \), and is independent of the selection of other types. This can be realized by generating customer types with random attributes.
- Estimate the expected number of arrivals \( \Lambda_i \) for any given customer type \( i \), by using historical data and possibly certain assumptions on customer preference.
- Estimate the total arrival rate \( \lambda(t) = \sum_{i=1}^{m} \lambda_i(t) \) of all customers at time \( t \). In practice, one can often use a discrete-time horizon, and then \( \lambda(t) \) is just the average number of arrivals in period \( t \).
Generate a random customer who arrives at time $t$. The probability that the customer is of type $i$ is $\lambda_i(t) / \lambda(t)$, which is the probability that an actual arrival at time $t$ belongs to type $i$. This can be easily achieved by drawing arrivals from data.

The algorithm simulates the derivative of benefit function $f'_j(t)$ in the following steps.

1. Select a random set $S \subseteq \{1, 2, \ldots, m\}$ of customer types. Every customer type has an equal probability to be chosen into $S$. Let $\epsilon = |S|/m$.

2. Estimate $\Lambda_i$ for each type $i \in S$.

3. Solve the following small LP

$$\begin{align*}
\text{max} & \quad \sum_{i \in S} \sum_{j=1}^n x_{ij} r_{ij} \\
\text{s.t.} & \quad \sum_{j=1}^n x_{ij} \leq \Lambda_i, \quad \text{for } i \in S \\
& \quad \sum_{i \in S} x_{ij} \leq \epsilon, \quad \text{for } j = 1, 2, \ldots, n \\
& \quad x_{ij} \geq 0.
\end{align*}$$

(19) (20) (21) (22)

Let $p = (p_1, p_2, \ldots, p_n)$ be the optimal dual variables corresponding to the constraints (21). Then we define a primal solution $x(p)$ to the original LP (2) as

$$x_{ij}(p) = 1_{ij}(p) \cdot \Lambda_i,$$

where

$$1_{ij}(p) = \begin{cases} 
1 & \text{if } j = \arg \max_k \{r_{ik} - p_k\} \text{ and } r_{ij} \geq p_j, \\
0 & \text{otherwise}. 
\end{cases}$$

We assume that there is no tie in determining the index $k$ that maximizes $r_{ik} - p_k$. This can be achieved by adding a small perturbation to the benefit values $r_{ij}$ (Feldman et al. 2010, Agrawal et al. 2014).

4. Generate a number of random arrivals at time $t$. Let $(b_1, b_2, \ldots, b_m)$ be the vector containing sample points of random arrivals, where $b_i$ is the number of arrival instances for type $i$ customers.

5. Estimate the total arrival rate $\lambda(t)$ at time $t$. 
Finally, \( f_j^*(t) \) is estimated by

\[
\hat{f}_j^*(t) = -\frac{\lambda(t)}{\|b\|_1} \sum_{i=1}^m b_i \cdot x_{ij}(p) / \Lambda_i \cdot (r_{ij} - f_j(t))^+ 
\]

(23)

\[
= -\frac{\lambda(t)}{\|b\|_1} \sum_{i=1}^m b_i \cdot 1_{ij}(p) \cdot (r_{ij} - f_j(t))^+. 
\]

(24)

Note that although the summation has \( m \) elements, at most \( \|b\|_1 \) of them are non-zero.

The idea of this simulation process is that when the dual prices \( p \) are approximately optimal for
the original LP (2), the induced primal solution \( x(p) \) will also be a near-optimal solution to the
LP (2).

The following result is first given by Feldman et al. (2010). Recall that \( \epsilon = |S|/m \). The result
says that when the number \( m \) of customer types is large, the above sampling procedure with a
fixed sample size \( |S| \) yields a solution \( x \), which is close in value to the optimal solution \( x^* \) with
high probability, as long as the average total arrival rate and the average relative total expected
benefit of each demand type is not too large.

Theorem 10. (Feldman et al. 2010) With high probability,

\[
\sum_{i=1}^m \sum_{j=1}^n r_{ij} x_{ij}(p) \geq (1 - O(\epsilon)) \sum_{i=1}^m \sum_{j=1}^n r_{ij} x_{ij}^*,
\]

given

\[
\max_{i,j} \left\{ \frac{r_{ij} \Lambda_i}{\sum_{k_l} r_{k_l} x_{k_l}^*} \right\} \leq \frac{\epsilon}{(n+1)(\ln m + \ln n)}
\]

and

\[
\max_i \{ \Lambda_i \} \leq \frac{\epsilon^3}{(n+1)(\ln m + \ln n)}.
\]

The following theorem guarantees that, if the primal solution \( x(p) \) used to compute the benefit
functions is near-optimal, then \( \mathbf{E}[\hat{f}_j^*(t)|S] \) performs well when used in our algorithms, where the
expectation is taken over the random sample \( (b_1, b_2, ..., b_m) \) of arrivals in step 4 above. According to
the central limit theorem, \( \hat{f}_j^*(t) \) converges to \( \mathbf{E}[\hat{f}_j^*(t)|S] \) when the sample size \( \|b\|_1 \) tends to infinity.

Thus, the number of samples \( \|b\|_1 \) should be chosen accordingly.
Theorem 11. Suppose $x(p)$ is $1 - O(\epsilon)$ optimal for the LP (2). If we use $E[\hat{f}_j(t)|S]$ as the derivative of the benefit function, our algorithms are $0.5(1 - O(\epsilon))$-competitive.

Proof. Let

$$g_j(t) \equiv -E[\hat{f}_j(t)|S] = \sum_{i=1}^{m} \lambda_i(t) \cdot 1_{ij}(p) \cdot (r_{ij} - \hat{f}_j(t))^+.$$ 

When we use $x(p)$ to route customers to resources in the Separation Algorithm, the optimal expected benefit for resource $j$ is just $\int_0^1 g_j(s)ds$. Then we can apply Theorem 3 to resource $j$ to get

$$\int_0^1 g_j(t)dt \geq 0.5 \sum_{i=1}^{m} r_{ij}x_{ij}(p),$$

which leads to

$$\sum_{j=1}^{n} \int_0^1 g_j(t)dt \geq 0.5 \sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij}x_{ij}(p).$$

Thus,

$$\sum_{j=1}^{n} \int_0^1 g_j(t)dt \geq 0.5 \sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij}x_{ij}(p) \geq 0.5(1 - O(\epsilon)) \sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij}x_{ij}^*.$$

This proves that our algorithms are $0.5(1 - O(\epsilon))$-competitive. □

8. Numerical Studies

We compare our Marginal Allocation Algorithm against the outcome of the actual scheduling practices used in the Division of Clinical Genetics within the Department of Pediatrics at Columbia University Medical Center (CUMC). The third author oversees appointment scheduling practice at the medical center. We estimate our model parameters, including patient preferences, arrival rates and hospital processing capacities, by using historical appointment-scheduling data from the outpatient clinics. We also test the performance of our algorithm against some simple heuristics. We find that our Marginal Allocation Algorithm performs the best among all heuristics considered, and is 21% more efficient than current practice, according to our performance metric, which we will explain below.

Specifically, we used data from the Division of Clinical Genetics at CUMC. Clinical Genetics is a field of medicine where adults are assessed for the risk of having offsprings with heritable conditions.
and children are assessed for genetic disorders. Geneticists use physical exams, chromosome testing and DNA analysis to diagnose patients suspected of having genetic abnormalities. The data we used contain more than 9000 appointment entries recorded in the year 2013. Each entry in the data records information about one appointment. The entry includes the date that the patient makes the appointment, the exact time of the appointment, whether the patient eventually showed up to the original appointment, canceled the appointment some time later, or missed the appointment. Canceled appointment slots are offered to new patients when possible.

The average number of patients who arrive to make appointments on each day is shown in Figure 1. During the week, there tend to be more patients who initiate requests for appointments on Thursday and Friday than on Monday and Tuesday. The actual arrival (of requests) pattern is highly non-stationary, as the average number of arrivals on Friday is about twice that on other days. Our Marginal Allocation Algorithm gracefully handles this inherent non-stationarity.

We assume that there are two sessions on each day, a morning and an afternoon session. Each session on each day corresponds to a resource in our model. About 98% appointments were scheduled into sessions on Monday through Thursday. We ignore the 2% of appointments scheduled into other sessions because there are insufficient data to perform accurate analysis for these sessions. In other words, we set the capacity of sessions on Friday, Saturday and Sunday to be 0. The capacity of sessions from Monday to Thursday are set based on the actual number of appointments made on these days, which is about 23 appointments per session. We will vary the capacity values in some of our experiments.

In this numerical experiment we do not model rescheduling, and treat each rescheduled appointment as an independent request. We also do not model the reuse of canceled appointment slots. Canceled slots are reused in practice, resulting in more efficient use of capacity. In this way, our algorithms are at a disadvantage compared to actual practice because it has less capacity at its disposal.

We assume that the higher the probability that a patient will show up for a session, the more preferred the session is. Thus, we use show probabilities as a proxy for patient preferences for each
Figure 1  Average number of arrivals in a week.

session in a week. Specifically, we define the benefit of assigning a patient who arrives in period $i$ to a session $j$ as

$$ r_{ij} = \text{Probability that the patient arriving in period } i \text{ will show up in session } j \text{ without canceling the appointment some time later or missing the appointment eventually.} \tag{25} $$

This definition of benefit value does not capture all practical concerns, but it gives a good sense of scheduling effectiveness. The higher the measure is, the fewer no-shows and cancellations are likely to result, and the fewer appointments slots are potentially wasted. The third author oversees appointment scheduling practice at CUMC. In practice, operators try to subjectively assign appointments to accommodate patient preferences while maintaining a high level of utilization of capacity. Because operators decisions are decentralized, they do not follow a precise and uniform procedure. However, our definition of benefit is compatible with the goals of the actual system.

We estimate the show probabilities as a function of 3 factors: the day of the week, the time of day (morning/afternoon) and the number of days of wait starting from the patient’s arrival to the actual appointment. In the first part of our experiment, we assume that patients have identical preferences in the sense that any two patients arriving on the same day will have the same benefit values for each open session. Thus, patients differ only in their time of arrival.

Both of the above assumptions regarding the homogeneity of preferences and the usefulness of show probabilities as indicators of preferences are strong assumptions. We are aware that the show probabilities are imperfect substitute for actual preferences. They also only express an average
measure of preference. A finer experiment would take into account actual preferences and variability of preferences among patients. However, we believe that our experiment is still valuable in indicating the value of using online algorithms. In a sense, our online algorithms are at a disadvantage compared to real practice because in practice, appointments were made taking into account actual preferences, whereas our online algorithms "know" only the show probabilities.

Figure 2 illustrates the show probabilities of patients who arrive on a Thursday to make appointments for the following week. We can see that, in general, the shorter the wait is in days, the higher the show probability is. Figure 3 illustrates the show probability as a function of number of days to wait before getting service. The show probabilities range from as low as 27%, for appointments made more than two months into the future, to as high as 97%, for same-day visits. Table 1 shows more show probabilities as a function of waiting time and day of week of the appointment.

Figure 2  Show probabilities of appointment slots assigned to patients who arrived on the previous Thursday.

Figure 3  Show probabilities as functions of number of days to wait before getting service.
Table 1  Show probabilities for morning sessions, as a function waiting time and day of week of the appointment. Some cells are NA because there is no patient arrival during weekends.

<table>
<thead>
<tr>
<th>Day of Week of Appointment</th>
<th>Number of days waiting</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Mon</td>
<td>91%</td>
</tr>
<tr>
<td>Tue</td>
<td>78%</td>
</tr>
<tr>
<td>Wed</td>
<td>97%</td>
</tr>
<tr>
<td>Thur</td>
<td>95%</td>
</tr>
</tbody>
</table>

We used a 12-week period from March to May in 2013 as our time horizon. An appointment reminder system was in use during this time. There are 2032 patients scheduled during this horizon according to our data. We use the sample consisting of these 2032 patient arrivals to simulate the performance of the following scheduling policies.

- The Marginal Allocation Algorithm (MAA). The arrival rates, which are inputs of the algorithm, are estimated using our one-year data in 2013. The average number of arrivals in each day of week has been shown earlier in Figure 1.

- The Marginal Allocation Algorithm with estimation error $\alpha\%$ (MAA-$\alpha\%$). This algorithm uses benefit values (25) that are each randomly and independently perturbed by relative errors drawn from a uniform distribution over $[-\alpha\%, \alpha\%]$. The total benefit earned by this algorithm is computed using the unperturbed benefit values. We include these algorithms to test the impact of our parameter estimation errors on the performance comparison with actual practice.

- The Separation Algorithm with larger than unit capacity.

- The outcome of actual practice used in hospitals. The total benefit earned by the actual strategy is also calculated using the benefit values defined in (25).

- The greedy policy, which always assigns a patient to the available session that is most preferred by the patient, as indicated by the show probability of the session. It captures a naive but easily implementable policy when a scheduler is aware of patient preferences.
The bid-price policy, which uses the optimal dual variables of LP (2) corresponding to the capacity constraints as the bid prices. It assigns an arriving customer to the resource with the lowest price smaller than or equal to the revenue that the customer brings. This heuristic is a widely used heuristic in resource-allocation problems.

In our first experiment, we do not consider overbooking and cancellations. The capacity of each session is set to be the number of appointments made in practice. In other words, we assume that the actual practice fully utilizes the capacity of all resources. Furthermore, we assume that patients arriving on the same day have homogeneous benefit values.

Since we use show probability as the benefit of scheduling a patient, the total benefit that a scheduling policy earns from the total 2032 patients is equal to the expected number of patients, among 2032, who will show up to the original appointments. In particular, since the show probabilities are themselves estimated based on the scheduling of the actual practice, the total benefit earned by the actual practice is just the actual number of patients, out of the total 2032, that showed up during the horizon.

For each scheduling policy, we report as its performance the ratio of total benefit to the total number 2032 of arrivals. This ratio represents the overall percentage of patients who will show up. Table 2 summarizes the performance of all scheduling policies we consider. We can see that our Marginal Allocation Algorithm performs the best, and in particular, gives more than 30% improvement over the actual practice, according to our performance measure. It is noteworthy that the greedy and bid-price policies do not have performance guarantees and can perform arbitrarily badly. In contrast, our Marginal Allocation Algorithm has not only a provable performance guarantee, but also good empirical performance.

The strength of our Marginal Allocation Algorithm is more directly reflected in comparison with the greedy policy. The greedy policy can be carried out by anyone as long as the patient preferences are exploited. Our Marginal Allocation Algorithm, which does smart reservation, gives 12.9% empirical improvement in scheduling efficiency over this heuristic. Note that in this experiment, all
Table 2  The empirical performance of different scheduling policies.

<table>
<thead>
<tr>
<th>Scheduling Policy</th>
<th>Performance of scheduling policies relative to LP upper bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual Strategy</td>
<td>67%</td>
</tr>
<tr>
<td>Greedy</td>
<td>81%</td>
</tr>
<tr>
<td>Bid-Price Heuristic</td>
<td>89%</td>
</tr>
<tr>
<td>Separation Algorithm</td>
<td>80%</td>
</tr>
<tr>
<td>MAA</td>
<td>92%</td>
</tr>
<tr>
<td>MAA-5%</td>
<td>91%</td>
</tr>
<tr>
<td>MAA-10%</td>
<td>88%</td>
</tr>
<tr>
<td>MAA-20%</td>
<td>83%</td>
</tr>
<tr>
<td>MAA-40%</td>
<td>74%</td>
</tr>
</tbody>
</table>

patients have the same priority. Our Marginal Allocation Algorithm is likely to exhibit much higher benefits when there are more patient types to consider because it can make more intelligent tradeoffs among the types than the greedy policy can. Remarkably, our Marginal Allocation Algorithm can be implemented as easily as the greedy policy. In the greedy approach, the scheduler has to be given a number representing estimated patient preference for each session. In our Marginal Allocation Algorithm, the scheduler also needs to be given only one number, namely the marginal value of benefit function, for each session.

8.1. Consideration for Overbooking

Starting from the numerical settings in the previous section, we study the practice of overbooking. Let $A_j$ be the actual number of patients who are assigned to session $j$. We assume that the actual strategy overbooks each session by a constant ratio, and thereby treat $C_j = \alpha A_j$ as the actual capacity of session $j$, where $\alpha \in [0, 1]$ is a scaling parameter that we vary in the numerical experiment.
We define the no-show probability as

\[ P_{NS} = \frac{\text{Total number of no-shows} + \text{Total number of appointments that are canceled no more than 2 days prior to the appointment time}}{\text{Total number of appointments}}. \]

The number is 26.89% as estimated from the data for Clinical Genetics.

A common practice is to take advantage of such high no-show probability by scheduling more patients to a session than its actual capacity can handle. Using terminology defined in Section 6.3, we use \( P_{NS} \) as the no-show probability for every session. We also vary the no-show penalty \( D \) in our experiments in the range \([2, 10]\). In this way, the pair \((\alpha, D)\) tunes the cost (17) of overbooking each session. The previous experiment corresponds to the case \( \alpha = 1, D = \infty \).

Now the total benefit of a scheduling policy is equal to the sum of all benefit values (25), i.e., show probabilities, earned from patients less the overbooking costs (17). In particular, we apply the function (17) of overbooking cost to the actual practice as well. That is, in our experiment the total overbooking costs incurred under the actual practice does not depend on the actual overbooked number of patients, but rather on the expected costs (17) estimated a priori. The performance of each scheduling policy is reported as its total benefit relative to the total benefit of the actual practice.

Table 3 summarizes the performance of scheduling policies when \( \alpha = 0.75 \) and \( D \) ranges from 2 to 15. Generally the performance of all policies decreases as the penalty \( D \) increases because of the reduced benefit of overbooking.

Table 4 reports the performance of scheduling policies when \( D = 3 \) and \( \alpha \) increases from 70% to 100%. The performance of all the scheduling policies reaches a limit for large values of \( \alpha \). This is because when \( \alpha \) is large, there is a large surplus of capacity associated with low overbooking costs. In such cases, scheduling policies virtually cannot see any capacity constraint, and thus have very good performance. Overall, for all values of \( \alpha \), our Marginal Allocation Algorithm performs at least 30% better than actual practice.
Table 3  The total benefit of scheduling policies relative to LP upper bound under different values of penalty $D$.

$\alpha = 0.75$.

<table>
<thead>
<tr>
<th>$D$</th>
<th>Actual Strategy</th>
<th>Greedy</th>
<th>Bid-Price Heuristic</th>
<th>Separation Alg.</th>
<th>MAA</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>70.1%</td>
<td>81.5%</td>
<td>89.0%</td>
<td>82.1%</td>
<td>93.2%</td>
</tr>
<tr>
<td>3</td>
<td>68.7%</td>
<td>80.6%</td>
<td>86.7%</td>
<td>82.0%</td>
<td>92.4%</td>
</tr>
<tr>
<td>4</td>
<td>66.8%</td>
<td>80.0%</td>
<td>86.6%</td>
<td>82.5%</td>
<td>92.3%</td>
</tr>
<tr>
<td>5</td>
<td>64.5%</td>
<td>79.5%</td>
<td>87.2%</td>
<td>82.8%</td>
<td>92.3%</td>
</tr>
<tr>
<td>6</td>
<td>62.2%</td>
<td>79.2%</td>
<td>88.7%</td>
<td>82.5%</td>
<td>92.0%</td>
</tr>
<tr>
<td>7</td>
<td>59.7%</td>
<td>78.9%</td>
<td>88.0%</td>
<td>82.6%</td>
<td>92.0%</td>
</tr>
<tr>
<td>8</td>
<td>57.1%</td>
<td>78.7%</td>
<td>88.6%</td>
<td>82.5%</td>
<td>92.0%</td>
</tr>
<tr>
<td>9</td>
<td>54.5%</td>
<td>78.2%</td>
<td>88.4%</td>
<td>82.4%</td>
<td>92.0%</td>
</tr>
<tr>
<td>10</td>
<td>51.8%</td>
<td>77.8%</td>
<td>88.4%</td>
<td>82.1%</td>
<td>91.5%</td>
</tr>
</tbody>
</table>

Table 4  The total benefit of scheduling policies relative to LP upper bound under different values of $\alpha$. $D = 3$.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>Actual Strategy</th>
<th>Greedy</th>
<th>Bid-Price Heuristic</th>
<th>Separation Alg.</th>
<th>MAA</th>
</tr>
</thead>
<tbody>
<tr>
<td>70%</td>
<td>62.7%</td>
<td>77.2%</td>
<td>88.3%</td>
<td>82.9%</td>
<td>92.2%</td>
</tr>
<tr>
<td>75%</td>
<td>68.7%</td>
<td>80.6%</td>
<td>86.7%</td>
<td>82.0%</td>
<td>92.4%</td>
</tr>
<tr>
<td>80%</td>
<td>70.8%</td>
<td>83.9%</td>
<td>88.9%</td>
<td>81.8%</td>
<td>93.4%</td>
</tr>
<tr>
<td>85%</td>
<td>71.4%</td>
<td>88.9%</td>
<td>92.5%</td>
<td>82.6%</td>
<td>94.5%</td>
</tr>
<tr>
<td>90%</td>
<td>71.1%</td>
<td>91.4%</td>
<td>94.4%</td>
<td>83.3%</td>
<td>94.8%</td>
</tr>
<tr>
<td>95%</td>
<td>70.7%</td>
<td>92.5%</td>
<td>95.9%</td>
<td>84.6%</td>
<td>95.8%</td>
</tr>
<tr>
<td>100%</td>
<td>70.5%</td>
<td>93.2%</td>
<td>95.7%</td>
<td>85.7%</td>
<td>96.2%</td>
</tr>
</tbody>
</table>

8.2. Consideration for Patient Availability

In the previous numerical experiments, patients who arrive in the same periods are treated as identical. However, in reality there is variability among patients’ availability. In this section, we capture this variability by simulating a particular chosen patient’s availability for a particular
session of the week as being drawn from a given distribution. This experiment tests whether more complex heterogeneous patient types affect the comparative performance of our algorithm.

We model the heterogeneity of patient availability as follows. A patient cannot be assigned to a session if he is unavailable for it. Otherwise, the benefit for the session is still the show probability as modeled in the previous sections. We assume that each patient has the same availability pattern for every week. A patient is available for any session with probability $P_A$, and this event is independent of the availability for other sessions in the same week. We vary $P_A$ from 15% to 100% to test the performance of all the scheduling policies we consider. When $P_A = 100\%$, the problem is reduced to the one in the last section, in which a patient can be assigned to any session.

Since we model 8 sessions in a week, one in the morning and one in the afternoon from Monday to Thursday (recall that there were very few appointments scheduled for Friday), each patient’s availability can be represented by an 8-dimension binary vector. Then, patients arriving in each period are further divided into 2$^8$ patient types, with $r_{i,k,j} = 0$ if a patient of type $k \in \{1, 2, ..., 2^8\}$ arriving in period $i$ is not available for session $j$.

We assume that the sessions offered by actual practice to patients were all available, so that the total benefit of actual practice is not affected by this newly modeled feature. The performance of each of the remaining scheduling policies is the averaged total benefit over 10,000 runs of simulation. In each simulation we draw the same 2032 number of arrivals from data, but we randomly generate patient availability. For $P_A$ ranging from 15% to 100%, Table 5 shows the performance of scheduling policies relative to the performance of actual practice. The relative performance is better for higher values of $P_A$, as there is more flexibility in scheduling when patients are available to more sessions. Even when $P_A$ is as small as 15%, our Marginal Allocation Algorithm still performs 8% better than actual practice. The gap gradually increases to more than 40% as $P_A$ increases.

9. Appendix

Proof of Theorem 3.
Table 5  The total benefit of scheduling policies relative to LP upper bound under different values of $P_A$, $D = 3$, $\alpha = 0.7$.

<table>
<thead>
<tr>
<th>$P_A$</th>
<th>Actual Strategy</th>
<th>Greedy</th>
<th>Bid-Price Heuristic</th>
<th>Separation Alg.</th>
<th>MAA</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.00%</td>
<td>88.8%</td>
<td>89.6%</td>
<td>96.0%</td>
<td>93.1%</td>
<td>96.4%</td>
</tr>
<tr>
<td>20.00%</td>
<td>77.6%</td>
<td>86.0%</td>
<td>93.5%</td>
<td>90.9%</td>
<td>95.1%</td>
</tr>
<tr>
<td>25.00%</td>
<td>72.5%</td>
<td>83.4%</td>
<td>92.1%</td>
<td>89.5%</td>
<td>94.2%</td>
</tr>
<tr>
<td>30.00%</td>
<td>70.0%</td>
<td>81.5%</td>
<td>91.4%</td>
<td>88.9%</td>
<td>93.7%</td>
</tr>
<tr>
<td>35.00%</td>
<td>68.4%</td>
<td>80.2%</td>
<td>91.4%</td>
<td>88.5%</td>
<td>93.5%</td>
</tr>
<tr>
<td>40.00%</td>
<td>67.2%</td>
<td>79.3%</td>
<td>90.8%</td>
<td>87.9%</td>
<td>93.2%</td>
</tr>
<tr>
<td>45.00%</td>
<td>66.3%</td>
<td>78.7%</td>
<td>91.5%</td>
<td>87.6%</td>
<td>93.0%</td>
</tr>
<tr>
<td>50.00%</td>
<td>65.7%</td>
<td>78.2%</td>
<td>90.7%</td>
<td>86.8%</td>
<td>92.7%</td>
</tr>
<tr>
<td>55.00%</td>
<td>65.1%</td>
<td>77.8%</td>
<td>90.5%</td>
<td>86.0%</td>
<td>92.7%</td>
</tr>
<tr>
<td>60.00%</td>
<td>64.7%</td>
<td>77.6%</td>
<td>89.3%</td>
<td>85.5%</td>
<td>92.7%</td>
</tr>
<tr>
<td>65.00%</td>
<td>64.3%</td>
<td>77.5%</td>
<td>88.7%</td>
<td>85.0%</td>
<td>92.8%</td>
</tr>
<tr>
<td>70.00%</td>
<td>64.0%</td>
<td>77.5%</td>
<td>88.9%</td>
<td>84.5%</td>
<td>92.5%</td>
</tr>
<tr>
<td>75.00%</td>
<td>63.7%</td>
<td>77.6%</td>
<td>88.6%</td>
<td>84.0%</td>
<td>92.4%</td>
</tr>
<tr>
<td>80.00%</td>
<td>63.4%</td>
<td>77.6%</td>
<td>87.7%</td>
<td>83.7%</td>
<td>92.3%</td>
</tr>
<tr>
<td>85.00%</td>
<td>63.2%</td>
<td>77.6%</td>
<td>88.3%</td>
<td>83.5%</td>
<td>92.4%</td>
</tr>
<tr>
<td>90.00%</td>
<td>63.0%</td>
<td>77.5%</td>
<td>88.4%</td>
<td>83.3%</td>
<td>92.5%</td>
</tr>
<tr>
<td>95.00%</td>
<td>62.9%</td>
<td>77.5%</td>
<td>88.0%</td>
<td>83.1%</td>
<td>92.4%</td>
</tr>
<tr>
<td>100.00%</td>
<td>62.7%</td>
<td>77.2%</td>
<td>88.3%</td>
<td>82.9%</td>
<td>92.2%</td>
</tr>
</tbody>
</table>

Proof. Consider a fixed resource $j$. Define $g(t) = -f_j'(t)$. Replacing the function $(\cdot)^+$ by inequalities, we can rewrite problem (6) as

\[
\inf_{g(t), \lambda_{ij}(t), z_i(t), r_{ij}, i=1,\ldots,m} \int_0^1 g(t)dt \\
\text{s.t. } g(t) = \sum_{i=1}^m \lambda_{ij}(t) z_i(t)
\]  

(26)
\[ g(t) \geq 0 \]
\[ z_i(t) \geq r_{ij} - \int_t^1 g(s) ds, \quad i = 1, 2, \ldots, m \quad \text{(28)} \]
\[ z_i(t) \geq 0, \quad i = 1, 2, \ldots, m \]
\[ \int_0^1 \sum_{i=1}^{m} \lambda_{ij}(t) dt \leq 1 \]
\[ \int_0^1 \sum_{i=1}^{m} \lambda_{ij}(t) r_{ij} dt = 1 \quad \text{(29)} \]
\[ \lambda_{ij}(t), r_{ij} \geq 0, \quad i = 1, 2, \ldots, m. \]

Note that because \( g(t) = \sum_{i=1}^{m} \lambda_{ij}(t) z_i(t) \) and the objective function minimizes \( \int_0^1 g(t) dt \), we want \( z_i(t) \) as small as possible. Therefore, the optimal solution must have \( z_i(t) = (r_{ij} - \int_t^1 g(s) ds)^+ \).

Then, we dualize the constraints (27), (28) and (29) using their Lagrangian multipliers, and obtain the following problem.

\[
\inf_{g(t), \lambda_{ij}(t), z_i(t), r_{ij}, i = 1, \ldots, m} g(t) \int_0^1 dt + \int_0^1 \gamma(t)[g(t) - \sum_{i=1}^{m} \lambda_{ij}(t) z_i(t)] dt \\
+ \sum_{i=1}^{m} \int_0^1 \theta_i(t)[r_{ij} - \int_t^1 g(s) ds - z_i(t)] dt \\
+ \omega\left(\int_0^1 \sum_{i=1}^{m} \lambda_{ij}(t) r_{ij} dt - 1\right)
\]

s.t. \( g(t) \geq 0 \)
\[ z_i(t) \geq 0, \quad i = 1, 2, \ldots, m \]
\[ \int_0^1 \sum_{i=1}^{m} \lambda_{ij}(t) dt \leq 1 \]
\[ \lambda_{ij}(t), r_{ij} \geq 0, \quad i = 1, 2, \ldots, m. \quad \text{(30)} \]

As long as \( \theta_i(t) \geq 0 \) for all \( t \in [0, 1] \) and \( i = 1, 2, \ldots, m \), the optimal objective value of problem (30) is a lower bound of (26), because every feasible solution of problem (26) is also feasible in (30), and the objective value of every feasible solution of problem (26) is greater than or equal to the corresponding objective value in (30). To find a lower bound on the optimal value of (26), we will instead find a lower bound on the optimal value of (30).
Next, we choose the following values for the dual variables.

\[ \gamma(t) = -0.5, \]

\[ \theta_i(t) = 0.5\lambda_{ij}(t), \text{ for } i = 1, 2, \ldots, m \]

\[ \omega = -0.5, \]

Since the constraint of (30) requires \( \lambda_{ij}(t) \geq 0 \), we have \( \theta_i(t) \geq 0 \). Thus, when using these values of dual variables, the optimal objective value of (30) is a lower bound of (26), and hence also a lower bound of problem (6). Plugging in these values, the objective function of (30) can be reduced to

\[
\int_0^1 g(t)dt - \int_0^1 0.5[g(t) - \sum_{i=1}^m \lambda_{ij}(t)z_i(t)]dt \\
+ \sum_{i=1}^m \int_0^1 0.5\lambda_{ij}(t)[r_{ij} - \int_t^1 g(s)ds - z_i(t)]dt \\
- 0.5(\int_0^1 \sum_{i=1}^m \lambda_{ij}(t)r_{ij}dt - 1) \\
= 0.5 \int_0^1 g(t)dt - 0.5 \sum_{i=1}^m \int_0^1 \lambda_{ij}(t) \left( \int_t^1 g(s)ds \right) dt + 0.5 \\
= 0.5 \int_0^1 g(t)dt - 0.5 \sum_{i=1}^m \int_0^t g(t) \left( \int_0^t \lambda_{ij}(s)ds \right) dt + 0.5 \\
= 0.5 \int_0^1 g(t) \left( 1 - \int_0^t \sum_{i=1}^m \lambda_{ij}(s)ds \right) dt + 0.5 \quad (31)
\]

Since we know from the constraint of (30) that \( \int_0^1 \sum_{i=1}^m \lambda_{ij}(t)dt \leq 1 \), we must have

\[ 1 - \int_0^1 \sum_{i=1}^m \lambda_{ij}(s)ds \geq 0 \]

for all \( t \leq 1 \). Thus, the infimum of (31) is achieved by setting \( g(t) = 0 \). The corresponding optimal objective value is 0.5. Therefore, 0.5 is a lower bound of problem (6). \( \square \)

Now that we have proved \( f_j(0)/\sum_{i=1}^m r_{ij}x_{ij}^* \geq 0.5 \) for every resource \( j \), we can readily show the competitive ratio of the Separation Algorithm.
Proof of Theorem 2. Since \( f_j(0) / \sum_{i=1}^{m} r_{ij}x^*_j \geq 0.5 \) for all \( j \), we must have

\[
\frac{\sum_{j=1}^{n} f_j(0)}{\sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij}x^*_j} \geq 0.5.
\]

From Theorem 1 we know that

\[
\sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij}x^*_j \geq E[\text{OPT}(\delta)],
\]

which gives

\[
\sum_{j=1}^{n} f_j(0) \geq 0.5E[\text{OPT}(\delta)].
\]

From Lemma 1 we know that \( \sum_{j=1}^{n} f_j(0) \) is the expected total benefit of the Separation Algorithm, so this algorithm is 0.5-competitive. \( \Box \).

Proof of Theorem 4.

Proof. Note that the distribution of \( \{ R(t) \}_{t \geq 0} \) is determined by the time points \( t_1, t_2, ..., t_{k-1} \).

In particular, for \( t \in (t_i, t_{i+1}) \), the value of \( P(R(t) = i) \) is only determined by \( t_1, t_2, ..., t_i \).

Given any value \( \beta \in (0, 1) \), we can construct a sequence of those time points \( t_1, t_2, ..., t_{k-1} \) recursively based on the following condition

\[
\int_{t_i}^{t_{i+1}} P(R(t) = i)dt = \frac{1}{\beta} - 1, \forall i = 0, 1, ..., k - 2.
\] (32)

Here \( t_i \) is when the barrier is increased to position \( i \), and is thus the first time that \( P(R(t) = i) \) becomes positive. Given \( t_1, t_2, ..., t_{i} \), this condition sets the value for \( t_{i+1} = t_{i+1}(\beta) \) by requiring that the area under the function \( P(R(t) = i) \) for \( t \in [t_i, t_{i+1}] \) is exactly \( 1/\beta - 1 \).

According to the above construction, since \( P(R(t) = i) \) is a continuous function of \( t \), the time points \( t_1, t_2, ..., t_{k-1} \) must change continuously in \( \beta \).

Furthermore, when \( \beta \to 1 \), i.e., the area under the function \( P(R(t) = i) \) for \( t \in [t_i, t_{i+1}] \) tends to 0 for each \( i = 0, 1, ..., k-2 \), we must have \( t_{i+1} - t_i \to 0 \) for each \( i = 0, 1, ..., i - 2 \). This implies that \( t_{k-1} \to t_0 = 0 \). On the other hand, when \( \beta \to 0 \), we have \( 1/\beta - 1 \to \infty \), so the area under \( P(R(t) = i) \) for \( t \in [t_i, t_{i+1}] \) can be arbitrarily large. In other words, by tuning the value of \( \beta \), we can set \( t_{k-1} \) to be any value within \( (0, k) \).
Therefore, there must exist some $\beta \in (0, 1)$ such that $t_{k-1}$ satisfies
\[
\int_{t_{k-1}}^{t_k} P(R(t) = k-1) dt = \frac{1}{\beta} - 1.
\]

Let $\beta^*$ be such a value that satisfies this condition. Set $\alpha^*_i(t) = P(R(t) = i)\beta^*$. We next prove that this construction of $\beta^*$ and $\alpha^*_i(t)$, for $i = 0, 1, ..., k-1$ and $t \in [0, k]$, satisfies the constraints of (15).

First of all, for $t \leq t_1$, we have
\[
\alpha^*_0(t) + \int_0^t \alpha^*_0(s) ds = \beta^* P(R(t) = 0) + \int_0^t \beta^* P(R(s) = 0) ds
\]
\[
= \beta^* \cdot 1 + \beta^* \int_0^t P(R(s) = 0) ds
\]
\[
\leq \beta^* + \beta^* \int_0^{t_1} P(R(s) = 0) ds
\]
\[
= \beta^* + \beta^*(1/\beta^* - 1)
\]
\[
= 1.
\]

Note that the inequality is tight when $t = t_1$. For $t > t_1$, since the barrier is above position 0, the value of the random process $R(t)$, when at position 0, is randomly jumping to position 1 at (hazard) rate equal to $P(R(t) = 0)$. Thus, we must have, for $t > t_1$,
\[
\frac{\partial P(R(t) = 0)}{\partial t} = -R(t)
\]
\[
\implies P(R(t) = 0) - P(R(t_1) = 0) = - \int_{t_1}^t P(R(s) = 0) ds
\]
\[
\implies P(R(t) = 0) + \int_0^t P(R(s) = 0) ds = P(R(t_1) = 0) + \int_0^{t_1} P(R(s) = 0) ds
\]
\[
\implies \alpha^*_0(t) + \int_0^t \alpha^*_0(s) ds = \alpha^*_0(t_1) + \int_0^{t_1} \alpha^*_0(s) ds = 1.
\]

Therefore, the first constraint of (15) holds and is tight for $t \geq t_1$.

To prove that the second constraint also holds, we recursively look at $i = 1, 2, ..., k-1$. Recall that $t_i$ is the first time that $P(R(t) = i)$ becomes positive. Thus for $t \leq t_i$ we have $P_i(R(t) = i) = 0$ and thus
\[
\alpha^*_i(t) + \int_0^t \alpha^*_i(s) ds = \beta^* P(R(t) = i) + \int_0^t \beta^* P(R(s) = i) ds = 0.
\]
For $t \in [t_i, t_{i+1}]$, we have
\[
\alpha_i^*(t) + \int_0^t \alpha_i^*(s)ds = \beta^* P(R(t) = i) + \int_0^t \beta^* P(R(s) = i)ds
\]
\[
= \beta^* P(R(t) = i) + \beta^* \int_{t_i}^{t_{i+1}} P(R(s) = i)ds
\]
\[
\leq \beta^* P(R(t) = i) + \beta^* \int_{t_i}^{t_{i+1}} P(R(s) = i)ds
\]
\[
= \beta^* P(R(t) = i) + \beta^* (1/\beta^* - 1)
\]
\[
= \beta^* P(R(t) = i) + \beta^* \int_{t_{i-1}}^{t_{i+1}} P(R(s) = i - 1)ds.
\]

When $t \in [t_i, t_{i+1}]$ and $R(t) = i$, the random process is actively bounded by the barrier, so the probability $P(R(t) = i)$, as a function of $t$, can only increase due to the transition from state $R(t) = i - 1$ to $R(t) = i$. The rate at which $P(R(t) = i)$ increases is $P(R(t) = i - 1)$. Thus, we have
\[
P(R(t) = i) = \int_{t_i}^{t_{i+1}} P(R(s) = i - 1)ds,
\]
which leads to
\[
\alpha_i^*(t) + \int_0^t \alpha_i^*(s)ds \leq \beta^* P(R(t) = i) + \beta^* \int_{t_{i-1}}^{t_{i+1}} P(R(s) = i - 1)ds
\]
\[
= \beta^* \int_{t_i}^{t_{i+1}} P(R(s) = i - 1)ds + \beta^* \int_{t_{i-1}}^{t_{i+1}} P(R(s) = i - 1)ds
\]
\[
= \beta^* \int_{t_i}^{t_{i+1}} P(R(s) = i - 1)ds
\]
\[
= \beta^* \int_{t_{i-1}}^{t_{i+1}} P(R(s) = i - 1)ds.
\]

Note that the inequality is tight when $t = t_{i+1}$.

For $t > t_{i+1}$, the barrier is above $i$, so the random process $R(t)$, if still at state $R(t) = i$, is not actively bounded by the barrier. Thus the state $R(t) = i$ is involved in two transitions: from state $i$ to $i + 1$, and from $i - 1$ to $i$. More precisely, we have for $t > t_{i+1},$
\[
\frac{\partial P(R(t) = i)}{\partial t} = P(R(t) = i - 1) - P(R(t) = i)
\]
\[
\Rightarrow P(R(t) = i) + \int_0^t P(R(s) = i)ds - \int_0^t P(R(s) = i - 1)ds = 0.
\]
$$\Rightarrow \alpha_i^*(t) + \int_0^t \alpha_i^*(s)ds = \int_0^t \alpha_{i-1}^*(s)ds. \quad (34)$$

This proves that the second constraint of (15) holds (and is tight for \( t \geq t_{i+1} \), for each \( i = 1, 2, ..., k - 1 \), respectively). Finally, the last constraint of (15) trivially holds because \( \sum_{i=0}^{k-1} P(R(t) = i) = 1 \Rightarrow \beta^* = \sum_{i=0}^{k-1} \alpha_i^*(t) \).

To prove (16), we can deduce that

$$\beta^* \sum_{i=0}^{k-1} i P(R(k) = i)$$

$$= \sum_{i=0}^{k-1} i \alpha_i^*(k)$$

$$= \sum_{i=0}^{k-1} i \left[ \int_0^k \alpha_{i-1}^*(s)ds - \int_0^k \alpha_i^*(s)ds \right] \quad \text{(by (34))}$$

$$= \sum_{i=0}^{k-1} \int_0^k \alpha_i^*(s)ds - k \int_0^k \alpha_{k-1}^*(s)ds \quad \text{(by canceling identical terms)}$$

$$= \int_0^k \left( \sum_{i=0}^{k-1} \alpha_i^*(s) \right) ds - k \beta^* \int_0^k P(R(s) = k - 1)ds$$

$$= \int_0^k \beta^* ds - k\beta^*(1/\beta^* - 1)$$

$$= 2k\beta^* - k.$$

We can then easily obtain (16) by rearranging terms. \( \square \)

**Proof of Lemma 2.**

**Proof.**

$$E[I_l|R(l) = l - j]$$

$$= \int_{l-1}^{l+1} P(R(s) = l|R(l) = l - j)ds$$

$$= \int_0^1 P_{\geq j}(s)ds.$$

Similarly, \( E[I_l|R(l) = l - j - 1] = \int_0^1 P_{\geq j+1}(s)ds \). Thus,

$$E[I_l|R(l) = l - j] - E[I_l|R(l) = l - j - 1]$$

$$= \int_0^1 P_{\geq j}(s)ds - \int_0^1 P_{\geq j+1}(s)ds$$
\[
\begin{align*}
&= \int_0^1 P_j(s)ds \\
&= \int_0^1 e^{-s \frac{q}{j}} ds \\
&= \sum_{\nu=j+1}^{\infty} e^{-1} \frac{1}{\nu!} \\
&= P_{\geq j+1}(1).
\end{align*}
\]

\[\square\]

**Proof of Lemma 3.**

*Proof.* Fix any \(i \in \{l, l+1, \ldots, k-1\}\). For ease of notation, define

\[\Delta_{i,j} \equiv E[I_i|R(d) = d - j] - E[I_i|R(d) = d - j - 1]\]

to be the decrease in the expected time that \(R(t)\) stays at the barrier during \([t_i, t_{i+1}]\), when the state at time \(t = d\) changes from \(R(d) = d - j\) down to \(R(d) = d - j - 1\), for all \(d = l, l+1, \ldots, i\) and \(j = 0, 1, \ldots, d - 1\).

From Lemma 2 we know that \(\Delta_{i,j} = P_{\geq j+1}(1)\). Furthermore, for \(d < i\) and \(d \geq l\), the value of \(E[I_i|R(d) = d - j]\) can be recursively computed by conditioning on \(R(d+1)\), i.e., on the movement of the random process during time \([d, d+1]\). Precisely,

\[E[I_i|R(d) = d - j] = \sum_{\nu=0}^{j} P_{\nu}(1) E[I_i|R(d+1) = d - j + \nu] + \sum_{\nu=j+1}^{\infty} P_{\nu}(1) E[I_i|R(d+1) = d].\]

Here, for example, \(R(d+1) = d - j + \nu\) represents the condition where the random process \(R(t)\) moves \(\nu\) steps upwards during time \([d, d+1]\); \(R(d+1) = d\) represents the condition where the random process hits the barrier at time \(t = d + 1\).

Similarly,

\[E[I_i|R(d) = d - j - 1] = \sum_{\nu=0}^{j+1} P_{\nu}(1) E[I_i|R(d+1) = d - j - 1 + \nu] + \sum_{\nu=j+2}^{\infty} P_{\nu}(1) E[I_i|R(d+1) = d] \]

\[= \sum_{\nu=0}^{j} P_{\nu}(1) E[I_i|R(d+1) = d - j - 1 + \nu] + \sum_{\nu=j+1}^{\infty} P_{\nu}(1) E[I_i|R(d+1) = d].\]
The above two equations lead to the following recursion for $\Delta_{d,j}$

\[
\Delta_{d,j} = E[I_i|R(d) = d - j] - E[I_i|R(d) = d - j - 1] \\
= \sum_{\nu=0}^{j} P_\nu(1)[E[I_i|R(d + 1) = d - j + \nu] - E[I_i|R(d + 1) = d - j - 1 + \nu]] \\
= \sum_{\nu=0}^{j} P_\nu(1)\Delta_{d+1,j-\nu+1}.
\] (35)

Note that in order to prove the lemma, we need to show $\Delta_{l,0}/\Delta_{l,1} \geq 1/e$. To this end, we prove a stronger result by constructing a bound on $\Delta_{d,j}/\Delta_{d,j+1}$ for all $d = l, l+1, \ldots, i$, and $j = 0, 1, \ldots, d$.

We construct the bounds using a sequence of ‘stationary’ values $\Delta_{*,0}, \Delta_{*,1}, \ldots$, which are defined based on the recursion (35) and are independent of $d$:

\[
\Delta_{*,0} = 1;
\]

\[
\Delta_{*,j} = \sum_{\nu=0}^{j} P_\nu(1)\Delta_{*,j-\nu+1}, \quad \forall j = 0, 1, 2, \ldots
\] (36)

\[
\Rightarrow \begin{cases} 
\Delta_{*,1} = e\Delta_{*,0}, \\
\Delta_{*,j+1} = (e-1)\Delta_{*,j} - \sum_{\nu=2}^{j} \frac{1}{\nu!} \Delta_{*,j-\nu+1}, \quad \forall j \geq 1.
\end{cases}
\] (37)

We next prove that

\[
\frac{\Delta_{d,j}}{\Delta_{d,j+1}} \geq \frac{\Delta_{*,j}}{\Delta_{*,j+1}}
\] (38)

using induction on $d$.

- First, we prove that (38) holds when $d = i$ by showing that $\Delta_{i,j}$ is decreasing in $j$ but $\Delta_{*,j}$ is increasing in $j$.

By Lemma 2, $\Delta_{i,j} = P_{\geq j+1}(1) > P_{\geq j+2}(1) = \Delta_{i,j+1}$, which means $\Delta_{i,j}$ is decreasing in $j$.

From (37) we know that $\Delta_{*,0} = e^{-1}\Delta_{*,1} < \Delta_{*,1}$. Provided that $\Delta_{*,\nu} \leq \Delta_{*,j}$ for all $\nu \leq j$ and some $j \geq 1$, we can deduce from (37),

\[
\frac{\Delta_{*,j+1}}{\Delta_{*,j}} \geq e - 1 - \sum_{\nu=2}^{j} \frac{1}{\nu!} \geq e - 1 - \sum_{\nu=2}^{\infty} \frac{1}{\nu!} = e - 1 - (e - 2) = 1.
\]

Thus, $\Delta_{*,j}$ is increasing in $j$, which finishes the proof that (38) holds when $d = i$. 
\[ \Delta_{d,j} \Delta_{\ast, j+1} - \Delta_{d,j+1} \Delta_{\ast, j} \]

\[ = \left( \sum_{\nu_1=0}^{j} P_{\nu_1}(1) \Delta_{d+1,j-\nu_1+1} \right) \left( \sum_{\nu_2=0}^{j+1} P_{\nu_2}(1) \Delta_{\ast,j-\nu_2+2} \right) \]

\[ - \left( \sum_{\nu_1=0}^{j} P_{\nu_1}(1) \Delta_{\ast,j-\nu_1+1} \right) \left( \sum_{\nu_2=0}^{j+1} P_{\nu_2}(1) \Delta_{d+1,j-\nu_2+2} \right) \]

(by (35) and (36))

\[ = \left( \sum_{\nu_1=0}^{j} P_{\nu_1}(1) \Delta_{d+1,j-\nu_1+1} \right) P_0(1) \Delta_{\ast,j+2} - \left( \sum_{\nu_1=0}^{j} P_{\nu_1}(1) \Delta_{\ast,j-\nu_1+1} \right) P_0(1) \Delta_{d+1,j+2} \]

\[ + \left( \sum_{\nu_1=0}^{j} P_{\nu_1}(1) \Delta_{d+1,j-\nu_1+1} \right) \left( \sum_{\nu_2=0}^{j} P_{\nu_2+1}(1) \Delta_{\ast,j-\nu_2+1} \right) \]

\[ - \left( \sum_{\nu_1=0}^{j} P_{\nu_1}(1) \Delta_{\ast,j-\nu_1+1} \right) \left( \sum_{\nu_2=0}^{j} P_{\nu_2+1}(1) \Delta_{d+1,j-\nu_2+1} \right) \]

\[ = \sum_{\nu_1=0}^{j} P_{\nu_1}(1) P_0(1) \left( \Delta_{d+1,j-\nu_1+1} \Delta_{\ast,j+2} - \Delta_{\ast,j-\nu_1+1} \Delta_{d+1,j+2} \right) \]

\[ + \sum_{\nu_1=0}^{j} \sum_{\nu_2=0}^{\nu_1-1} P_{\nu_1}(1) P_{\nu_2+1}(1) \left( \Delta_{d+1,j-\nu_1+1} \Delta_{\ast,j-\nu_2+1} - \Delta_{\ast,j-\nu_1+1} \Delta_{d+1,j-\nu_2+1} \right) \]

\[ + \sum_{\nu_1=0}^{j} \sum_{\nu_2=0}^{\nu_1-1} P_{\nu_1}(1) P_{\nu_2+1}(1) \left( \Delta_{d+1,j-\nu_1+1} \Delta_{\ast,j-\nu_2+1} - \Delta_{\ast,j-\nu_1+1} \Delta_{d+1,j-\nu_2+1} \right) \]

\[ + \sum_{\nu_1=0}^{j} \sum_{\nu_2=0}^{\nu_1-1} P_{\nu_1}(1) P_{\nu_2+1}(1) \left( \Delta_{d+1,j-\nu_1+1} \Delta_{\ast,j-\nu_2+1} - \Delta_{\ast,j-\nu_1+1} \Delta_{d+1,j-\nu_2+1} \right) \]

\[ = \sum_{\nu_1=0}^{j} P_{\nu_1}(1) P_0(1) \left( \Delta_{d+1,j-\nu_1+1} \Delta_{\ast,j+2} - \Delta_{\ast,j-\nu_1+1} \Delta_{d+1,j+2} \right) \]

\[ + \sum_{\nu_1=0}^{j} \sum_{\nu_2=0}^{\nu_1-1} P_{\nu_1}(1) P_{\nu_2+1}(1) \left( \Delta_{d+1,j-\nu_1+1} \Delta_{\ast,j-\nu_2+1} - \Delta_{\ast,j-\nu_1+1} \Delta_{d+1,j-\nu_2+1} \right) \]

\[ + \sum_{\nu_1=0}^{j} \sum_{\nu_2=0}^{\nu_1-1} \left( P_{\nu_1}(1) P_{\nu_2+1}(1) - P_{\nu_2}(1) P_{\nu_1+1}(1) \right) \left( \Delta_{d+1,j-\nu_1+1} \Delta_{\ast,j-\nu_2+1} - \Delta_{\ast,j-\nu_1+1} \Delta_{d+1,j-\nu_2+1} \right) \]

(39)

(40)
Now using induction on $d + 1$, we know that for $\nu_1 \geq 0$,

$$\frac{\Delta_{d+1,j-\nu_1+1}}{\Delta_{d+1,j+2}} \geq \frac{\Delta_{*,j-\nu_1+1}}{\Delta_{*,j+2}},$$

and thus $(39) \geq 0$. In $(40)$, since $\nu_1 > \nu_2$, we can use induction on $d + 1$ again to obtain

$$\frac{\Delta_{d+1,j-\nu_1+1}}{\Delta_{d+1,j-\nu_2+1}} \geq \frac{\Delta_{*,j-\nu_1+1}}{\Delta_{*,j-\nu_2+1}},$$

which implies

$$\Delta_{d+1,j-\nu_1+1} \Delta_{*,j-\nu_2+1} - \Delta_{*,j-\nu_1+1} \Delta_{d+1,j-\nu_2+1} \geq 0.$$

Furthermore, since $\nu_1 > \nu_2$,

$$P_{\nu_1}(1)P_{\nu_2+1}(1) - P_{\nu_2}(1)P_{\nu_2+1}(1) = P_{\nu_1}(1)P_{\nu_2}(1) \left( \frac{1}{\nu_2 + 1} - \frac{1}{\nu_1 + 1} \right) \geq 0.$$

Thus, $(40) \geq 0$. In sum, we have shown $\Delta_j \Delta_{*,j+1} - \Delta_{*,j+1} \Delta_{*,j} \geq 0$, which finishes the induction proof for condition $(38)$.

This result $(38)$ directly leads to the statement of the Lemma

$$\Delta_{l,0} \geq \Delta_{l,1} \frac{\Delta_{*,0}}{\Delta_{*,1}} = \Delta_{l,1}/e.$$

□

Proof of Lemma 4.

Proof. Fix any $i \in \{l, l+1, ..., k-2\}$. By symmetry, we have $E[I_i|R(l) = l] = E[I_{l+1}|R(l+1) = l+1]$ and $E[I_i|R(l) = l-1] = E[I_{l+1}|R(l+1) = l]$.

Thus,

$$E[I_i|R(l) = l] - E[I_{l+1}|R(l) = l] - (E[I_i|R(l) = l-1] - E[I_{l+1}|R(l) = l-1])$$

$$= E[I_{l+1}|R(l+1) = l+1] - E[I_{l+1}|R(l+1) = l] - (E[I_{l+1}|R(l) = l] - E[I_{l+1}|R(l) = l-1])$$

$$= E[I_{l+1}|R(l+1) = l+1] - E[I_{l+1}|R(l+1) = l] - e^{-1}(E[I_{l+1}|R(l+1) = l] - E[I_{l+1}|R(l+1) = l-1])$$

$$\geq 0,$$

where the last inequality follows from Lemma 3. □
Proof of Lemma 6.

Proof. Given any sequence of times points \( \bar{t}_1 \leq \bar{t}_2 \leq \cdots \leq \bar{t}_t \) such that \( \bar{t}_j \leq j \), \( \forall j = 1, 2, ..., l \), we want to prove the lemma when \( t_j = \bar{t}_j \) for \( j \leq l \) and \( t_j = j \) for \( j > l \).

Fix any \( i \in \{l, l + 1, ..., k - 2\} \). Initially, set \( t_j = j \) for all \( j = 1, 2, ..., k - 1 \), and we know that \( E[I_i] \geq E[I_{i+1}] \) by Lemma 5. We next prove that \( E[I_i] \geq E[I_{i+1}] \) always holds when we reduce the value of \( t_j \) from \( j \) to \( \bar{t}_j \) sequentially for \( j = 1, 2, ..., l \).

Define

\[
I'_i = \begin{cases} 
I_i, & \text{if } i > l \\ 
\int_{t_i}^{t_{i+1}} 1(R(s) = l)ds, & \text{if } i = l.
\end{cases}
\]

By this definition, \( I_i \) is different from \( I'_i \) only if \( i = l \) and \( t_i < l \) (we create the definition of \( I'_i \) so as to facilitate the proof for the case of \( i = l \)). Note that we always have \( I_i \geq I'_i \), and in particular, \( I_i = I'_i \) when initially \( t_j = j \) for all \( j = 1, 2, ..., l \).

Consider the result of reducing \( t_j \) from \( j \) to \( \bar{t}_j \), when \( t_d = d \) for all \( d = j + 1, j + 2, ..., k - 1 \). Let \( \hat{R}(t), \hat{I}, \text{ and } \hat{I}' \) be the value of \( R(t), I_i \) and \( I'_i \), respectively, before reducing \( t_j \), i.e., when \( t_j = j \). Let \( \bar{R}(t), \bar{I}_i \) and \( \bar{I}'_i \) be the new value of \( R(t), I_i \) and \( I'_i \), respectively, after reducing \( t_j \) to \( \bar{t}_j \).

Suppose \( E[I_i] \geq E[I'_{i+1}] \) holds, we next prove that \( E[I_i] \geq E[I'_{i+1}] \) also holds.

- When \( t_j = j \), since \( j \leq l \), we must have \( t_i = l \) and thus \( E[I_i] = E[I'_i] \).
- We must have \( P(\hat{R}(j) = \nu) = P(\bar{R}(j) = \nu) \) for all \( \nu \leq j - 2 \), because if \( R(t) \leq j - 2 \), the random process is not affected by the barrier after time \( t_{j-1} \).
- We must have \( P(\hat{R}(j) = j - 1) \leq P(\bar{R}(j) = j - 1) \) and \( P(\hat{R}(j) = j) \geq P(\bar{R}(j) = j) \) because when \( t_j \) becomes smaller, there is more time for the random process \( R(t) \) to jump from state \( j - 1 \) up to \( j \). Moreover,

\[
P(\hat{R}(j) = j) - P(\hat{R}(j) = j - 1) = P(\bar{R}(j) = j - 1) - P(\bar{R}(j) = j - 1) \geq 0. \quad (41)
\]

Based on the above results, we can then deduce that \( E[I_i] \) is defined as \( E[I_i] \) given \( t_j = \bar{t}_j \). Similarly, \( E[I'_i] \) is defined as \( E[I'_i] \) given \( t_j = j \)

\[
E[I_i] - E[I'_{i+1}]
\]
Now Lemma 4 gives

\[ \frac{\nu}{E[\hat{t}_i] - E[\hat{t}_{i+1}]} \]

\[ = \sum_{\nu=0}^{j} E[\hat{t}_i | \hat{R}(j) = \nu] P(\hat{R}(j) = \nu) - \sum_{\nu=0}^{j} E[\hat{t}_{i+1} | \hat{R}(j) = \nu] P(\hat{R}(j) = \nu) \]

\[ = \sum_{\nu=0}^{j} E[\hat{t}_i | \hat{R}(j) = \nu] P(\hat{R}(j) = \nu) - \sum_{\nu=0}^{j} E[\hat{t}_{i+1} | \hat{R}(j) = \nu] P(\hat{R}(j) = \nu) \]

\[ \geq 0 \]

\( \text{Given } R(j) = \nu, \text{ reducing } t_j \text{ does not affect the random process after } t \geq j, \)

\[ \text{due to the memoryless property.} \]

\[ = \sum_{\nu=0}^{j-2} E[\hat{t}_i | \hat{R}(j) = \nu] P(\hat{R}(j) = \nu) + \sum_{\nu=j-1}^{j} E[\hat{t}_i | \hat{R}(j) = \nu] P(\hat{R}(j) = \nu) \]

\[ \text{−} \sum_{\nu=0}^{j-2} E[\hat{t}_{i+1} | \hat{R}(j) = \nu] P(\hat{R}(j) = \nu) - \sum_{\nu=j-1}^{j} E[\hat{t}_{i+1} | \hat{R}(j) = \nu] P(\hat{R}(j) = \nu) \]

\[ = E[\hat{t}_i] + \sum_{\nu=j-1}^{j} E[\hat{t}_i | \hat{R}(j) = \nu](P(\hat{R}(j) = \nu) - P(\hat{R}(j) = \nu)) \]

\[ \text{−} E[\hat{t}_{i+1}] - \sum_{\nu=j-1}^{j} E[\hat{t}_{i+1} | \hat{R}(j) = \nu](P(\hat{R}(j) = \nu) - P(\hat{R}(j) = \nu)) \]

\[ = E[\hat{t}_i] - E[\hat{t}_{i+1}] + \sum_{\nu=j-1}^{j} (E[\hat{t}_i | \hat{R}(j) = \nu] - E[\hat{t}_{i+1} | \hat{R}(j) = \nu])(P(\hat{R}(j) = \nu) - P(\hat{R}(j) = \nu)) \]

\[ = E[\hat{t}_i] - E[\hat{t}_{i+1}] + \sum_{\nu=j-1}^{j} (E[\hat{t}_i | \hat{R}(j) = \nu] - E[\hat{t}_{i+1} | \hat{R}(j) = \nu])(P(\hat{R}(j) = \nu) - P(\hat{R}(j) = \nu)) \]

\[ \geq 0 \]

\( \text{(by (41))} \)

Now Lemma 4 gives

\[ E[\hat{t}_i | \hat{R}(j) = j] - E[\hat{t}_{i+1} | \hat{R}(j) = j] - E[\hat{t}_i | \hat{R}(j) = j - 1] + E[\hat{t}_{i+1} | \hat{R}(j) = j - 1] \geq 0. \]
This proves that $E[I_j] \geq E[I_{j+1}]$. Therefore, we always have $E[I_j] \geq E[I_{j+1}]$ when we change $t_j$ from $j$ to $\bar{t}_j$ for all $j = 1, 2, ..., l$. □

**Proof of Theorem 5.**

**Proof.** We give a new and more detailed construction of the same set of times points as constructed in the proof of Theorem 4.

Fix any $\beta \in (0, 1)$. Starting with $t_i = i$, $\forall i = 1, 2, ..., k - 1$, we run the following algorithm:

For $i = 0, 1, ..., k - 2$:

(a) If $E[I_i] > 1/\beta - 1$, reduce $t_{i+1}$ such that $E[I_i] = 1/\beta - 1$.

(b) Stop if $E[I_i] < 1/\beta - 1$.

If the algorithm stops at (b) when $i = l$, we must have

$$E[I_0] = E[I_1] = \cdots = E[I_{l-1}] = 1/\beta - 1$$

and, according to Lemma 6,

$$1/\beta - 1 = E[I_{l-1}] \geq E[I_l] \geq \cdots \geq E[I_{k-1}]. \quad (42)$$

On the other hand, if the algorithm never stops at (b), we must have

$$E[I_0] = E[I_1] = \cdots = E[I_{k-2}] = 1/\beta - 1. \quad (43)$$

If we change the value of $\beta$, the time points $t_1, t_2, ..., t_{k-1}$ as the result of the algorithm must change continuously in $\beta$. This implies that $E[I_{k-1}] = \int_{t_{k-1}}^k P(R(s) = k-1) ds$ must change continuously in $\beta$. When $\beta$ is close to 0, we must have $E[I_{k-1}] < 1/\beta - 1$; when $\beta$ is close to 1, we must have $E[I_{k-1}] > 1/\beta - 1$. Therefore, there must exist a $\beta$ such that, when the algorithm ends,

$$E[I_{k-1}] = 1/\beta - 1.$$ 

Let $\beta^*$ be such a value.

Now the time points have met all desired conditions if (43) holds (i.e., the algorithm never stops at (b)). If the algorithm stops at some step (b), then according to (42),

$$1/\beta^* - 1 = E[I_{l-1}] \geq E[I_l] \geq \cdots \geq E[I_{k-1}] = 1/\beta^* - 1$$
\[ \Rightarrow \mathbb{E}[I_0] = \mathbb{E}[I_1] = \cdots = \mathbb{E}[I_{k-1}] = 1/\beta^* - 1, \]

which gives all desired conditions of the time points as well. \(\square\)

**Lemma 8.**

\[ \beta^* = \frac{1}{2} + \frac{1}{2k} \sum_{i=0}^{k-1} i \alpha_i^*(k). \]

**Proof.** Starting from Theorem 4 we can deduce that

\[ k(1 - \beta^*) = \beta^* \left[ \sum_{i=0}^{k-1} i P(R(k) = i) \right] \]

\[ \Rightarrow 2k\beta^* = k + \beta^* \sum_{i=0}^{k-1} i P(R(k) = i) = k + \sum_{i=0}^{k-1} i \alpha_i^*(k) \]

\[ \Rightarrow \beta^* = \frac{1}{2} + \frac{1}{2k} \sum_{i=0}^{k-1} i \alpha_i^*(k). \]

\(\square\)

**Proof of Lemma 7.**

**Proof.** If suffices to prove the case when \(x = y + 1\). We have

\[ \sum_{i=-l}^{l} P_{k-1+i-(y+1)}(\lambda) - \sum_{i=-l}^{l} P_{k-1+i-y}(\lambda) \]

\[ = P_{k-2-l-y}(\lambda) - P_{k-1+l-y}(\lambda). \]

If \(k - 2 - l - y < 0\), the lemma trivially holds because \(P_{k-2-l-y}(\lambda) = 0\) and thus \(P_{k-2-l-y}(\lambda) - P_{k-1+l-y}(\lambda) \leq 0\).

Now suppose \(k - 2 - l - y \geq 0\). Then

\[ \frac{P_{k-2-l-y}(\lambda)}{P_{k-1+l-y}(\lambda)} = \frac{\lambda^{k-2-l-y}}{(k - 2 - l - y)!} \frac{(k - 1 + l - y)!}{\lambda^{k-1+l-y}} \]

\[ = \frac{1}{\lambda^{2l+1}} \prod_{i=-l}^{l} (k - 1 - y + i). \]

Since \(y \geq k - 1 - \lambda\), we must have \(k - 1 - y \leq \lambda\) and \((k - 1 - y + i)(k - 1 - y - i) \leq \lambda^2\). This shows that \(P_{k-2-l-y}(\lambda)/P_{k-1+l-y}(\lambda) \leq 1\) and thus \(P_{k-2-l-y}(\lambda) - P_{k-1+l-y}(\lambda) \leq 0\). \(\square\)
Lemma 9. For any \( l = 0, 1, ..., k - 1 \), we have

\[
\sum_{i=-l}^{l} P_{k-1+i}(k) \leq \frac{1}{\beta^*} \sum_{i=0}^{l} \alpha_{k-1-i}^*(k).
\]

Proof. Define

\[
R^{(i)} \equiv R(t_i) + N(k) - N(t_i), \quad \forall i = 0, 1, 2, ..., k.
\]

Note that since the bounded process \( R(t) \) is determined by \( N(t) \), the random variables \( R^{(i)} \)'s are also determined by \( N(t) \).

Since

\[
P(R(0) = i) = P(R(t_0) + N(k) - N(t_0) = i) = P(N(k) = i) = P_i(k)
\]

and

\[
P(R(k) = i) = P(R(k) = i) = \alpha_i^*(k)/\beta^*,
\]

it suffices to show that

\[
\sum_{i=-l}^{l} P(R^{(j-1)} = k - 1 + i) \leq \sum_{i=-l}^{l} P(R^{(j)} = k - 1 + i) \tag{44}
\]

for all \( l = 0, 1, ..., k - 1 \) and \( j = 1, 2, ..., k \).

According to the definition of the bounded process \( R(t) \), if \( R(t_{j-1}) + N(t_j) - N(t_{j-1}) \leq j - 1 \), then \( R(t_j) = R(t_{j-1}) + N(t_j) - N(t_{j-1}) \) and thus

\[
R^{(j)} = R(t_j) + N(k) - N(t_j)
\]

\[
= R(t_{j-1}) + N(t_j) - N(t_{j-1}) + N(k) - N(t_j)
\]

\[
= R(t_{j-1}) + N(k) - N(t_{j-1})
\]

\[
= R^{(j-1)}.
\]

Therefore,

\[
\sum_{i=-l}^{l} P(R^{(j-1)} = k - 1 + i|R(t_{j-1}) + N(t_j) - N(t_{j-1}) \leq j - 1)
\]

\[
= \sum_{i=-l}^{l} P(R^{(j)} = k - 1 + i|R(t_{j-1}) + N(t_j) - N(t_{j-1}) \leq j - 1)
\]
for all $l = 0, 1, \ldots, k - 1$ and $j = 1, 2, \ldots, k$.

Now consider the case $x = R(t_{j-1}) + N(t_j) - N(t_{j-1}) > j - 1$. We must have

$$R^{(j-1)} = x + N(k) - N(t_j),$$

$$R^{(j)} = j - 1 + N(k) - N(t_j).$$

Recall that Lemma 5 gives $t_j \leq j$, so $x > j - 1 \geq t_j - 1 = k - 1 - (k - t_j)$. We can then apply Lemma 7 by further setting $y = j - 1$ and $\lambda = k - t_j$ and obtain (for $x > j - 1$)

$$\sum_{i=-l}^{l} P(R^{(j-1)} = k - 1 + i|R(t_{j-1}) + N(t_j) - N(t_{j-1}) = x)$$

$$= \sum_{i=-l}^{l} P(x + N(k) - N(t_j) = k - 1 + i|R(t_{j-1}) + N(t_j) - N(t_{j-1}) = x)$$

$$= \sum_{i=-l}^{l} P(N(k) - N(t_j) = k - 1 + i - x|R(t_{j-1}) + N(t_j) - N(t_{j-1}) = x)$$

$$= \sum_{i=-l}^{l} P(N(k - t_j) = k - 1 + i - x)$$

$$\leq \sum_{i=-l}^{l} P(N(k - t_j) = k - 1 + i - (j - 1))$$

$$= \sum_{i=-l}^{l} P(j - 1 + N(k) - N(t_j) = k - 1 + i|R(t_{j-1}) + N(t_j) - N(t_{j-1}) = x)$$

$$= \sum_{i=-l}^{l} P(R^{(j)} = k - 1 + i|R(t_{j-1}) + N(t_j) - N(t_{j-1}) = x)$$

In sum, we have shown (44), which proves the lemma. □

**Proof of Theorem 6.**

*Proof.* Combining Lemma 8 and Lemma 9, we obtain

$$\beta^* = \frac{1}{2} + \frac{1}{2k} \sum_{i=0}^{k-1} i \alpha_i^*(k)$$

$$= \frac{1}{2} + \frac{1}{2k} \sum_{i=0}^{k-2} \sum_{i=0}^{l} \alpha_{k-1-i}^*(k)$$

$$\geq \frac{1}{2} + \frac{1}{2k} \sum_{i=0}^{k-2} \sum_{i=-l}^{l} \beta^* P_{k-i+i}(k)$$
\[= \frac{1}{2} + \frac{\beta^*}{2k} \left[ \sum_{l=0}^{k-2} \sum_{i=-l}^{0} P_{k-1+i}(k) + \sum_{l=0}^{k-2} \sum_{i=1}^{l} P_{k-1+i}(k) \right] \]

\[= \frac{1}{2} + \frac{\beta^*}{2k} \left[ \sum_{i=1}^{k-1} iP_i(k) + \sum_{i=k}^{2k-2} (2k-i)P_i(k) \right] \]

\[= \frac{1}{2} + \frac{\beta^*}{2k} \left[ \sum_{i=1}^{k-1} iP_i(k) + \sum_{i=k}^{2k-2} (2k-2i)P_i(k) \right] \]

\[= \frac{1}{2} + \frac{\beta^*}{2k} \left[ \sum_{i=1}^{k-1} iP_i(k) - 2 \sum_{i=1}^{k-1} iP_{k-1+i}(k) \right]. \]

\[\Rightarrow \beta^* \geq \frac{k}{2k - \left[ \sum_{i=1}^{2k-2} iP_i(k) - 2 \sum_{i=1}^{k-1} iP_{k-1+i}(k) \right]} = \frac{k}{k + \left[ k - \sum_{i=1}^{2k-2} iP_i(k) \right] + 2 \sum_{i=1}^{k-1} iP_{k-1+i}(k)} = \frac{k}{k + \sum_{i=2k-2}^{\infty} iP_i(k) + 2 \sum_{i=1}^{k-1} iP_{k-1+i}(k)} \]

\[= 1 + \frac{1}{k} \left[ \sum_{i=2k-1}^{\infty} iP_i(k) + 2 \sum_{i=1}^{k-1} iP_{k-1+i}(k) \right]. \]

**Proof of Corollary 1.**

**Proof.**

\[\sum_{i=2k-1}^{\infty} iP_i(k) + 2 \sum_{i=1}^{k-1} iP_{k+i-1}(k) \leq \sum_{i=1}^{2k-1} iP_i(k) \]

\[= 2 \sum_{i=1}^{\infty} iP_i(k) - k \frac{(k+i-1)!}{(k-1)!} \leq 2 \left[ \frac{1}{(k-1)!} \frac{e^{-k} k^k}{k!} + \sum_{i=k}^{\infty} P_i(k) \right]. \]

\[\Rightarrow \beta^* \geq \frac{1}{1 + \frac{1}{k} \left[ \frac{1}{(k-1)!} \frac{e^{-k} k^k}{k!} + \sum_{i=k}^{\infty} P_i(k) \right]} = \frac{1}{1 + \frac{1}{k} \left[ \frac{e^{-k} k^k}{k!} + \frac{P_{\frac{k^2}{k}}(k)}{k} \right]} . \]

By Stirling’s formula,

\[\frac{e^{-k} k^k}{k!} = \frac{1}{\sqrt{2\pi k}} + o(1/k). \]
Furthermore, since $P_{\geq k}(k) \leq 1$, we have

$$\frac{P_{\geq k}(k)}{k} = O(1/k).$$

Thus,

$$\frac{1}{1 + 2 \left[ \frac{-k}{k^2} + \frac{P_{\geq k}(k)}{k} \right]} = \frac{1}{1 + 2 \left[ \frac{1}{\sqrt{2\pi k}} + o(1/k) + O(1/k) \right]} = 1 - \sqrt{\frac{2}{\pi}} \cdot \frac{1}{\sqrt{k}} + O(1/k).$$

In sum, we have proved that

$$\frac{V_0(0)}{\int_0^1 r(t)\lambda(t)dt} \geq \beta^* \geq 1 - \sqrt{\frac{2}{\pi}} \cdot \frac{1}{\sqrt{k}} + O(1/k).$$

Thus, if $C_j \geq k$ for all resource $j$,

$$\sum_{i=1}^n f_j(t, C_j) = \frac{V_0(0)}{\sum_{i=1}^n r_{ij}} \geq 1 - \sqrt{\frac{2}{\pi}} \cdot \frac{1}{\sqrt{k}} + O(1/k)$$

$$\implies \sum_{i=1}^n \sum_{j=1}^n x_{ij} r_{ij} \geq 1 - \sqrt{\frac{2}{\pi}} \cdot \frac{1}{\sqrt{k}} + O(1/k).$$

This proves the competitive ratio of the Separation Algorithm. $\Box$

**Proof of Theorem 7.**

*Proof.* Let $h(t, c)$ be the expected future benefit that can be obtained by the Marginal Allocation Algorithm starting at time $t$ when $c = (c_1, c_2, ..., c_n)$ is the vector of resources available at $t$. Let $f(t, c) = \sum_{j=1}^n f_j(t, c_j)$ be the expected future benefit that can be obtained by the Separation Algorithm starting at time $t$ when $c$ is the vector of resources available at $t$. We will show that

$$h(t, c) \geq f(t, c) \quad (45)$$

for every given state $(t, c)$.

Define an algorithm $\Pi^{(i)}$ as follows. For the first $i$ customers, apply the Marginal Allocation Algorithm. Afterward, for the $(i+1)$-th, $(i+2)$-th, ..., customers, apply the Separation Algorithm. Let $h^{(i)}(t, c)$ be the expected future benefit when algorithm $\Pi^{(i)}$ is applied starting at time $t$ with remaining inventory $c$, and assuming that no customers have arrived prior to time $t$. We must have

$$h^{(0)}(t, c) = f(t, c),$$

This proves the competitive ratio of the Separation Algorithm. $\Box$
\[
\lim_{i \to \infty} h^{(i)}(t, c) = h(t, c).
\]

The HJB equation for computing the expected benefit of algorithm \(\Pi^{(1)}\) is

\[
\frac{\partial h^{(1)}(t, c)}{\partial t} = -\sum_{i=1}^{m} \lambda_i(t) \left( \max_{j \in \{1, 2, ..., n\}} (r_{ij} - f_j(t, c_j) + f_j(t, c_j - 1))^+ - \Delta^{(1)}(t, c) \right),
\]

where

\[
\Delta^{(1)}(t, c) \equiv h^{(1)}(t, c) - f(t, c).
\]

The boundary conditions for (46) are \(h^{(1)}(1, c) = 0\) (the length of the horizon is 1) and \(f_j(t, -1) = -\infty, \forall j = 1, 2, ..., n\) and \(t \in [0, 1]\). Note that according to the boundary condition \(f_j(t, -1) = -\infty\), the ‘bid price’ of any resource \(j\) that has no remaining inventory becomes infinity, as \(f_j(t, 0) - f_j(t, -1) = \infty\).

To see why (46) is true, consider the discrete version of (46). During any small period \((t, t + \delta t)\), one of the following three events will take place.

- No customer arrives during \((t, t + \delta t)\). Then the expected future benefit \(h^{(1)}(t, c)\) turns into \(h^{(1)}(t + \delta t, c)\).

- A customer of some type \(i\) arrives, but \(\Pi^{(1)}\) (which applies the Marginal Allocation Algorithm to the customer) rejects the customer. We must have

\[
\max_{j \in \{1, 2, ..., n\}} (r_{ij} - f_j(t + \delta t, c_j) + f_j(t + \delta t, c_j - 1))^+ = 0,
\]

because \(r_{ij}\) must be smaller than the ‘bid price’ \(f_j(t + \delta t, c_j) - f_j(t + \delta t, c_j - 1)\) of all available resources \(j\); we must also have that \(h^{(1)}(t, c)\) turns into \(f(t + \delta t, c)\), as \(\Pi^{(1)}\) turns into the Separation Algorithm.

- A customer of some type \(i\) arrives and the customer is assigned to a resource \(j\). In this case, the system collects benefit \(r_{ij}\), and \(h^{(1)}(t, c)\) turns into \(f(t + \delta t, c - e_j)\) as \(\Pi^{(1)}\) turns into the Separation Algorithm, where \(e_j\) is the unit vector with the \(j\)-th position being 1. Note that

\[
f(t + \delta t, c - e_j) = f(t + \delta t, c) + f_j(t + \delta t, c_j - 1) - f_j(t + \delta t, c_j).
\]
Then mathematically, we can combine the second and the third bullet points, and say that when a customer of type \( i \) arrives, the expected future benefit \( h^{(1)}(t, c) \) turns into total current and future benefit

\[
f(t + \delta t, c) + \max_{j \in \{1, 2, \ldots, n\}} (r_{ij} - f_j(t + \delta t, c_j) + f_j(t + \delta t, c_j - 1))^+.
\]

In sum, the recursive equation for \( h^{(1)}(t, c) \) is

\[
h^{(1)}(t, c) = (1 - \sum_{i=1}^{m} \lambda_i(t) \delta t) h^{(1)}(t + \delta t, c) + \sum_{i=1}^{m} \lambda_i(t) \delta t \left( f(t + \delta t, c) + \max_{j \in \{1, 2, \ldots, n\}} (r_{ij} - f_j(t + \delta t, c_j) + f_j(t + \delta t, c_j - 1))^+ \right).
\]

Letting \( \delta t \to 0 \) leads to (46).

Therefore,

\[
\frac{\partial h^{(1)}(t, c)}{\partial t} \leq \frac{\partial f(t, c)}{\partial t} + \sum_{i=1}^{m} \lambda_i(t) \Delta^{(1)}(t, c). \tag{47}
\]

This equation implies that, if at some time \( t_0 \) we have \( \Delta^{(1)}(t_0, c) < 0 \) or equivalently

\[
h^{(1)}(t_0, c) - f(t_0, c) < 0, \tag{48}
\]

then we must have

\[
\frac{\partial h^{(1)}(t, c)}{\partial t} < \frac{\partial f(t, c)}{\partial t}, \quad \forall t \in (t_0, 1] \tag{49}
\]

and

\[
h^{(1)}(t, c) < f(t, c), \quad \forall t \in (t_0, 1]. \tag{50}
\]

However, since we know that \( h^{(1)}(1, c) = f(1, c) = 0 \), (50) cannot be true, and thus (48) cannot be true. Therefore, we have proved

\[
h^{(1)}(t, c) \geq f(t, c), \quad \forall t \in [0, 1]. \tag{51}
\]

Next, we show that

\[
h^{(i)}(t, c) \geq h^{(i-1)}(t, c), \quad \forall t \in [0, 1] \tag{52}
\]
by induction on $i$.

Equation (51) already proves the base case $i = 1$. Suppose for some $\bar{i} > 1$, (52) holds for all $i < \bar{i}$. Now we show that it also holds for $i = \bar{i}$. By definition, for any $\bar{i} > 1$, algorithms $\Pi^{(\bar{i})}$ and $\Pi^{(\bar{i} - 1)}$ must allocate the first customer in the same way. Thus, $\Pi^{(\bar{i})}$ and $\Pi^{(\bar{i} - 1)}$ earn the same benefit from the first customer, and then transit into the same state. After that first customer, $\Pi^{(\bar{i})}$ continues to apply $\Pi^{(\bar{i} - 1)}$ pretending that no customer has ever arrived, while $\Pi^{(\bar{i} - 1)}$ continues to apply $\Pi^{(\bar{i} - 2)}$. By induction, the expected future benefit of $\Pi^{(\bar{i} - 1)}$ is at least that of $\Pi^{(\bar{i} - 2)}$. Therefore, the expected future benefit of $\Pi^{(\bar{i})}$ is at least that of $\Pi^{(\bar{i} - 1)}$.

Thus, we have proved (52). It immediately follows that

$$h^{(\infty)}(t,c) \geq h^{(0)}(t,c).$$

\[\square\]

**Proof of Theorem 8.**

*Proof.* It suffices to prove that for each $j$,

$$\lim_{\theta \to \infty} \frac{f_j(0, C_j)}{\sum_{i=1}^{m} r_{ij} x^*_ij} = \lim_{\theta \to \infty} \frac{f_j(0, \theta \bar{C}_j)}{\theta \sum_{i=1}^{m} r_{ij} \bar{x}^*_ij} = 1.$$

Consider the single-resource benefit-maximization problem for resource $j$. Let $N$ be the total number of customers who will come to resource $j$. $N$ is a Poisson random variable with mean

$$\mathbb{E}[N] = \sum_{i=1}^{m} x^*_ij = \theta \sum_{i=1}^{m} \bar{x}^*_ij.$$

Let $M = \max_{i,j} r_{ij}$ be an upper bound on all of the benefits. A first-come, first-served algorithm for resource $j$ will admit $\min(N, C_j)$ customers, and obtain an expected total benefit of at least

$$\sum_{i=1}^{m} x^*_ij r_{ij} - \mathbb{E}[N - \min(N, C_j)]M,$$

where $x^*_ij r_{ij}$ is the total expected benefit from all type $i$ customers, and $N - \min(N, C_j)$ is the number of customers who cannot fit into the $C_j$ units of capacity.
Since the dynamic programming algorithm must perform at least as well as the first-in, first-served algorithm, we have

\[ f_j(0, C_j) \geq \sum_{i=1}^{m} x_{ij}^* r_{ij} - E[N - \min(N, C_j)] M. \]

When \( \theta \) goes to infinity, both \( x^* \) and \( E[N] \) grow in proportion to \( \theta \). Thus,

\[
\lim_{\theta \to \infty} \frac{f_j(0, C_j)}{\sum_{i=1}^{m} r_{ij} x_{ij}^*} \geq \lim_{\theta \to \infty} \frac{\sum_{i=1}^{m} x_{ij}^* r_{ij} - E[N - \min(N, C_j)] M}{\sum_{i=1}^{m} r_{ij} x_{ij}^*} \\
\geq 1 - \lim_{\theta \to \infty} \frac{E[N - \min(N, \sum_{i=1}^{m} x_{ij}^*)] M}{\sum_{i=1}^{m} r_{ij} x_{ij}^*} \\
\geq 1 - \lim_{\theta \to \infty} \frac{0.4 \sqrt{\sum_{i=1}^{m} x_{ij}^*} M}{\sum_{i=1}^{m} r_{ij} x_{ij}^*} \\
= 1 - \lim_{\theta \to \infty} \frac{\sqrt{\theta}}{\theta} \cdot \frac{0.4 \sqrt{\sum_{i=1}^{m} x_{ij}^*} M}{\sum_{i=1}^{m} r_{ij} x_{ij}^*} \\
= 1,
\]

where the last inequality uses the fact that, when the mean of Poisson random variable \( N \) is very large, \( E[(N - E[N])^+] \) is bounded by \( 0.4 \sqrt{E[N]} \). \( \square \)

**References**


Qin, Chao, Huanan Zhang, Cheng Hua, Cong Shi. 2015. A simple admission control policy for revenue management problems with non-stationary customer arrivals. working paper.


