Submitted to *Operations Research*
manuscript (Please, provide the manuscript number!)

Multi-priority Online Scheduling with Cancellations

Xinshang Wang, Van-Anh Truong
Department of Industrial Engineering and Operations Research, Columbia University, New York, NY 10027, USA,
xw2230@columbia.edu, vatruong@ieor.columbia.edu

We study a fundamental model of resource allocation in which a finite amount of service capacity must be allocated to a stream of jobs of different priorities arriving randomly over time. Jobs incur costs and may also cancel while waiting for service. To increase the rate of service, overtime capacity can be used at a cost. This model has application in healthcare scheduling, server applications, make-to-order manufacturing systems, general service systems, and green computing. We present an online algorithm that minimizes the total cost due to waiting, cancellations and overtime capacity usage. We prove that our scheduling algorithm has cost at most twice of an optimal offline algorithm. This competitive ratio is the best possible for this class of problems. We also provide extensive numerical experiments to test the performance of our algorithm and its variants.

Key words: Analysis of algorithms, Approximations/heuristic, Cost analysis

1. Introduction

In many applications, a finite amount of a service resource must be allocated to a stream of jobs arriving randomly over time. Jobs are prioritized based on certain criteria such as profitability or urgency. When immediate service is not available, arriving jobs join a priority queue to be served at a later time. While waiting, jobs may cancel their requests and leave the queue randomly. A *cancellation* is any job that expires or leaves the system without being processed. To increase the rate of service, overtime resource can be used at a higher cost. The system must dynamically

determine the service rate that minimizes the total cost due to waiting, cancellations and overtime resource usage.

The above problem is central to many applications in Operations Research. For example, in healthcare facilities, jobs correspond to patient requests for resources such as diagnostic devices and operating rooms. Patients are often prioritized based on their urgency and served in order of priority (Min and Yih 2010). Longer wait times, which result in lower quality of care, are represented as a cost on the system. Time is often slotted. With a limited number of time slots available each day, only a certain number of patients can be served on each day; the remaining patients must join a waitlist (Denton et al. 2010, Ayvaz and Huh 2010, Gerchak et al. 1996). Patients in the waitlist may randomly cancel their requests, thus leaving the system. Often, patients can be served using surge capacity or overtime (Patrick et al. 2008) at an additional cost. The scheduler must select the number of patients to serve each day, using surge capacity or overtime as needed, to minimize the total cost, including waiting costs, lost revenue due to cancellations, and the cost of overtime work.

Another application is network routing for server applications. In this setting, jobs correspond to data requests sent to server applications by local clients. In order to process jobs, servers make use of computing resources such as CPU and disk I/O. These finite resources limit the rate of service. Arriving data requests that cannot be immediately processed are stored in the memory. If the wait time is too long, some data requests might expire or lose the value of being processed, thus leaving the queue. For instance, in some applications, data requests are sent with time-outs and need not be processed after their delay exceeds the time-out (Xiong et al. 2008); in others, data contain information that gradually loses value over time such as the location of a mobile device. The priority of a data request is determined by its expiration date or the probability of expiring. In the case of server congestion, data packages can often be routed to remote (external) idle servers at a cost of propagation delay (Lin et al. 2012); such routing increases the service rate temporarily. The routing decision needs to be dynamically made to reduce the overall cost generated from data expiration, local congestion and processing delays due to data routing.

In many service systems, make-to-order manufacturing systems, retail stores and call centers, jobs correspond to customers arriving randomly over time. Depending on the application, customers may be served in order of their priorities. Backlogged customers may cancel their orders (Rubino and Ata 2009, Blackburn 1972), resulting in lost sales and even 'reshelving' costs (Martin et al. 1992). In these settings, the service rate can often be increased by using overtime work (Dellaert and Melo 1998, Özdamar and Yazgaç 1997), on-call workers (Greenhouse 2012), or expedited procurement of parts. These strategies have the effect of temporarily increasing the service rate at an increased variable cost. The manager needs to dynamically determine the service policy so as to control the total system cost.

In the scaling of computer processing speed for minimizing energy usage (Bansal et al. 2009a, Yao et al. 1995), jobs correspond to sequences of CPU instructions that arrive randomly. Jobs are often prioritized and processed in order of priority. With recent technologies, the processing speeds of CPUs can be dynamically raised at the cost of a higher rate of power usage. Such a speed-scaling technique often helps to save more power than the simple strategy of turning off a device during idle periods. The goal is to minimize the sum of some measure of quality of service, such as job completion time and total energy consumption (Bansal et al. 2009a).

Our model captures most, if not all, of these applications. Specifically, we consider a discrete-time planning horizon of T periods, where T is possibly infinite. Jobs are categorized into n priority classes. Each class is associated with a waiting cost, a cancellation probability and a cancellation cost. Jobs are either processed in the current period or are added to a priority queue. In each period t, a number C_t of jobs of any priority can be processed. Additional jobs can be processed at an extra variable cost.

The above scheduling problem is especially difficult to analyze in real applications due to the difficulty in forecasting future information. On the demand side, future arrivals are often class-dependent and time-dependent (Huh et al. 2013), which requires an enormous amount of data to estimate the joint distribution of demand for multiple classes. For instance, a patient request

often leads to subsequent periodic requests, resulting in the time correlation of demand. Also, in markets of new products or services, demand is often driven by intensive promotion campaigns, in which case future demand depends on promotional and social factors and is highly uncertain. On the supply side, processing capacities are often subject to occasional failures such as staff absenteeism, machine breakdown (Federgruen and So 1990) and server crashes, which can be very hard to predict.

Even with access to accurate joint distributions of future demand and supply, the computation of an optimal scheduling policy is often intractable. When there are t jobs in the system, it takes $O(t^n)$ space to store all possible states in a given period. This difficulty is referred to as *the curse of dimensionality*.

In view of these difficulties, we aim to develop near-optimal scheduling policies that are robust to future information and are easy to compute. In this paper, we make no assumptions about the joint distribution of future arrivals and capacities. Instead, we study an online version of the problem. A problem is *online* if at all points in time, exogenous future information is completely unknown and the algorithm has to make adaptive decisions based on past and current information. In contrast, an *offline* algorithm knows all future information up-front. *Competitive Analysis* is the most widely used method for evaluating online algorithms (Borodin and El-Yaniv 1998). It considers the relative performance between an online algorithm and an optimal offline algorithm under the worst input instance. The maximum ratio between the cost achieved under the online algorithm and that under the optimal offline algorithm is called the *competitive ratio* for that online algorithm. An algorithm with a competitive ratio of α is said to be α*-competitive*.

For the scheduling problem without cancellations, we propose 2-competitive randomized and deterministic online algorithms. For the scheduling problem with cancellations, we relax the assumption of the online problem by making the 'offline' policy unaware of which jobs will cancel, i.e., the random cancellation events are exogenous to both the online and offline policies. Under this definition, we propose 2-competitive online algorithms for the model with cancellations. Further, we show that the competitive ratio of our deterministic algorithm is the best that can be achieved.

Our proofs of the competitive ratios use a cost-balancing approach in conjunction with the following new ideas.

- We construct a novel *distance function* which summarizes in a single number the difference between the history of the online algorithm OLN and the optimal offline algorithm OFF. The distance function $\phi_t(\text{OLN}, \text{OFF})$ has a nice physical interpretation. At any time t, if we immediately service $\phi_t(\text{OLN}, \text{OFF})$ additional jobs under the online algorithm, the remaining jobs will have lower priorities than the current remaining jobs under the offline algorithm. The distance function dynamically accounts for the difference in the number of scheduled and cancelled jobs between the two algorithms.
- Depending on the sign of the distance function in each period, we *partition* the periods in the planning horizon into two sets. We show that in each type of the periods, one cost component of the online algorithm is dominated by the corresponding component of the offline algorithm. This result naturally leads to the proof of the competitive ratios.
- For the model with cancellations, we use *stochastic coupling* to compare the exogenous cancellation events under the online and offline algorithms. When extended to the model with cancellations, our distance function incorporates the difference in the number of coupled cancellation events between the two algorithms.
- For the model with cancellations, we propose a new *cost-accounting scheme* which transforms cancellation costs into new waiting and overtime costs. This transformation allows the algorithms for the model without cancellations to be easily extended to capture cancellation behaviours.

2. Literature Review

Our work is related to the literature on Appointment Scheduling, which has been studied intensively. For comprehensive reviews of the broader area, see Guerriero and Guido (2011), May et al. (2011), Cardoen et al. (2010) and Gupta (2007). A large part of the literature considers *intra-day* scheduling. In these problems, the number of patients to be served on each day is given or is exogenous, and the task is to set the sequence and the start time of each appointment so as to control

patient wait time and provider idle time. Another part of the literature models *multi-day* scheduling. In these problems, the allocation of patients to days is dynamically controlled. Some of this literature allows patients to be scheduled into future days at the time of arrival. This paradigm is called *advance scheduling*. See, for example, Truong (2014b), Gocgun and Ghate (2012) and Patrick et al. (2008). In the rest of the multi-day literature, only the number of patients to be scheduled to the current period is determined. The rest of the patients are added to a waitlist. This paradigm is called *allocation scheduling*. See, for example, Huh et al. (2013), Min and Yih (2010), Ayvaz and Huh (2010) and Gerchak et al. (1996). So far, very few works have studied the optimal advance-scheduling policy. Recently, Truong (2014b) linked the solutions for the advance and allocation scheduling problems by showing that for a two-class model, their optimal scheduling polices are equivalent. This result points to the importance of allocation scheduling as a fundamental model.

Our model is an allocation-scheduling model. In allocation scheduling, past works have used dynamic programming to explore structural properties of the optimal scheduling policy. When there are one or two patient classes, the problem is easy to solve. For multi-class problems, some structural results are known but there is no policy with performance guarantees. Gerchak et al. (1996) and Huh et al. (2013) study scheduling problems with two patient classes. Patients in the emergent class require same-day service; patients in the elective class can wait. Gerchak et al. (1996) show that the optimal scheduling policy is not a cut-off policy; the optimal number of admissions increases in the size of waitlist. Huh et al. (2013) develop heuristics for a correlated and dynamic environment. Min and Yih (2010) and Ayvaz and Huh (2010) study the allocation-scheduling problem with multiple elective patient classes. Min and Yih (2010) develop bounds on the optimal number of admissions. They show that priority-based discrimination results in as much as a 30% difference in the optimal number of admissions compared to an undiscriminated scheme. Ayvaz and Huh (2010) analyze the structural properties of an optimal scheduling policy and study numerical performance of a protect-constant heuristic. The heuristics presented in these works do not come with any performance guarantees. Moreover, for the static policies that they propose,

such as the protect-constant policies, it is easy to search for the best protect-constant levels only when the number of demand classes is small. When the number of demand classes is large, it is much harder to search for the best set of protect-constant levels without additional structural properties. Thus, in multi-class settings, even heuristics with good empirical performance are hard to find.

The scheduling system we consider is related to make-to-order manufacturing systems in that processing capacity is used to service realized demand. These make-to-order systems are usually modeled as queuing systems. In the framework of queuing systems, service times and inter-arrival times must be stationary, independent and most often, exponentially distributed to ensure that the model is tractable. Our approach differs from this literature in that we do not assume any joint distribution on future arrivals and service capacities. For reviews on admission control for make-to-order queues, see Stidham (1985) and more recently, Carr and Duenyas (2000). Blackburn (1972) studies the optimal strategies for turning on or off a server subject to reneging customers. Their work is related to ours in that they consider the dynamic expansion of the service rate. While they only consider one type of jobs, we allow jobs to have multiple priorities, each with a different cancellation probability. Rubino and Ata (2009) consider a related problem in which customers can be outsourced and have chances to renege. They propose a heuristic based on the solution to the problem in the heavy-traffic regime.

Our model is related to the work of Keskinocak et al. (2001), who study single-server online scheduling problems with lead-time quotation, with application to make-to-order manufacturing systems. In their model, jobs can be rejected upon arrival, and waiting costs are incurred in each period before the jobs are finished. The rejection of jobs is similar to the use of overtime resource in our model. Our work can be seen as a multi-priority, multi-server extension of their model, and with further considerations for job reneging. We note that in their model, a job may span multiple periods, while in our model, every job can be finished in a single period. However, our model easily accommodates batched arrivals. A job that takes multiple periods to finish can be modelled as a batched arrival.

The class of machine and multiprocessor scheduling problems share some characteristics with our work. In a typical machine-scheduling problem, jobs must be assigned to one or more machines so as to minimize a chosen objective such as the makespan, the total completion time or the total waiting time. Our model resembles a machine-scheduling problem in which (1) jobs have unit processing times, (2) jobs can be rejected or diverted after being released, (3) each job has a specific release time, which is used to define a waiting cost, and (4) jobs may cancel randomly. However, an online version of this model has not been considered. Overtime usage and job cancellations are not common in the machine-scheduling literature. We refer the reader to Chen et al. (1998) for a detailed survey of machine scheduling. Among the existing literature, the most relevant works include Noga and Seiden (2001) and Zhang et al. (2009). Noga and Seiden (2001) consider an online machine-scheduling problem where jobs have release times and the objective is to minimize the total waiting cost, but the service rate cannot be dynamically controlled. Zhang et al. (2009) study a deterministic offline scheduling problem where jobs can be rejected.

Our work is related to speed scaling problems in the management of power for a single processor. In these problems, CPU processing speeds can be raised by supplying more power. One group of works considers the optimization problem of some energy related objective, subject to deadlines for job completion (Yao et al. 1995, Chan et al. 2007, Bansal et al. 2007a, 2009b, 2011). The first theoretical study of such model is given by Yao et al. (1995). They show that an optimal offline algorithm for any convex power function can be computed by a greedy method. They also give an online algorithm with constant competitive ratio when the power function is polynomial. Another group of works considers energy usage and job waiting time (Albers and Fujiwara 2007, Bansal et al. 2007b, 2009a). Bansal et al. (2009a) propose an online algorithm that minimizes the sum of fractional waiting costs and energy usage for arbitrary power functions. When there are no cancellations, our model captures the tradeoff in Bansal et al. (2009a). In the literature of speed scaling, most works consider continuous-time models. Our model captures a discrete-time speed scaling problem in which processing speeds can only be changed in discrete periods. Cancellation behaviours are generally not considered in the speed-scaling literature.

Our online algorithms and their performance guarantees are related to many other approximation algorithms developed in Operations Management. Approximation algorithms have performance guarantees that are relative to the optimal stochastic dynamic policy, while online algorithms have performance guarantees that are relative to an optimal offline algorithm. The latter type of guarantee is much stronger. Moreover, in order to use approximation algorithms it is still necessary to estimate the joint distribution of future arrivals. These estimates can be very hard to make. In contrast, the online algorithms we study do not have this requirement. Levi et al. (2005) propose a cost-balancing technique for inventory control problems. They prove that this cost-balancing algorithm is a 2-approximation. The cost-balancing technique is found to be very adaptable and is applied in approximation algorithms for many other supply-chain problems (see, for example, Levi et al. (2008a) and Levi et al. (2008b)). Recently, Truong (2014a) develops an approximation algorithm for the stochastic inventory control problem by using a look-ahead optimization approach.

Many online algorithms have been developed recently for problems in Operations Management. Ball and Queyranne (2009) consider an online version of a revenue management problem. They show that the simple protection-level policy gives the best possible competitive ratio. The ratio depends on the level of price discounts. Wagner (2010) considers the online economic lot-sizing problem. They model the online profit-maximizing problem as a min-max game, and provide conditions under which the competitive ratio is bounded. Buchbinder et al. (2013) study an online algorithm for a make-to-order variant of the joint-replenishment problem for which they proved a competitive ratio of three. Elmachtoub and Levi (2014) study a general class of customer-selection problems where decisions are made in two phases: In the first phase, arriving customers with different configurations are selected in an online manner. Then in the second phase, the cost of the service system is generated based on the set of selected customers. They develop a framework of analysis for this class of problems and apply it to various models.

The remainder of this paper is organized as follows. In Section 3, we present our online algorithm for the scheduling model without cancellations. In Section 3.3, we generalize our results to the

case that future costs are discounted. In Section 3.4, we discuss lower bounds on the competitive ratio and prove that our algorithm is optimal. In Section 4, we extend the model and algorithm to capture cancellations. Finally, in Section 5, we report the numerical performance of our scheduling policies.

3. Model of Allocation Scheduling without Cancellations

The planning horizon has T periods, indexed from 1 to T, where T may be infinite. There are n priority classes. Each class i is associated with a waiting cost $w_i \geq 0$, which is incurred when a class i job stays in the waitlist for one period. Let the n classes be ordered in decreasing order of priority. We assume that the waiting costs satisfy $w_1 \geq w_2 \geq \cdots \geq w_n$. The scheduling policies we present in the paper do not depend on the total number n of classes, so n can be arbitrarily large and the collection of waiting costs can even approach a continuous distribution.

At the beginning of each period t, we observe the vector $s_t = (s_{t1}, s_{t2}, ..., s_{tn})$ representing the total number of jobs currently in the waitlist, where s_{ti} is the number of jobs in class i. Then, we observe the regular capacity C_t, which is the number of jobs, regardless of priority, that can be processed by regular resource in period t. Next we observe the number of new arrivals $\delta_t = (\delta_{t1}, \delta_{t2}, ..., \delta_{tn})$, where δ_{ti} stands for the number of arrivals of class i jobs. We have $s_t, \delta_t \in \mathbb{Z}_+^n$ and $C_t \in \mathbb{Z}_+$, where $\mathbb{Z}_+$ is the set of all non-negative integers. For an online algorithm, C_t and δ_t are completely unknown until period t, while for an offline algorithm the entire sample path $\{(C_t, \delta_t)\}_{t=1,2,...,T}$ is known at the beginning of period 1.

After the new arrivals have occurred, the number of jobs in system is represented by the vector $s_t + \delta_t$. From among the $\|s_t + \delta_t\|_1$ jobs in system, a scheduling policy determines the number $a_t \in \mathbb{Z}_+$ of jobs to service in period t. It is intuitive that once the number a_t is decided, it is optimal to serve the a_t jobs with the highest priorities. This property is proved in Ayvaz and Huh (2010) and Min and Yih (2010) for optimal stochastic policies. The same result holds here. However, we do not repeat the proof. We restrict our attention to the class of policies that follow this service scheme.

If $a_t > C_t$, we assume that the additional $a_t - C_t$ jobs will be served by overtime resource incurring a total *overtime cost* of $(a_t - C_t)p$, where p is the cost of using an *overtime slot*. If $a_t \leq C_t$, no overtime cost will be incurred. Define $d_t \equiv (a_t - C_t)^+$ as the number of overtime slots used in period t. We normalize all cost values such that $p = 1$. Then the total overtime cost in period t is just d_t.

Because we only consider policies that schedule some number of highest priority jobs in each period, two scheduling policies differ only in the timing and number of jobs drawn from the wait-list. We introduce the following operator that extracts a certain number of jobs with the highest priorities from a given system state $s_t \in \mathbb{Z}_+^n$.

DEFINITION 1. For a vector $x \in \mathbb{Z}_+^n$ and a non-negative integer k, we define $h(x,k) \in \mathbb{Z}_+^n$ as the vector that contains the k jobs with the highest priorities in x. Let $h_i(x,k)$ be the ith element of $h(x,k)$. Let $h(x,k) = 0$ for $k < 0$, and $h(x,k) = x$ for $k > \|x\|_1$.

Since the w_i's are decreasing in i, we have for $0 \leq k \leq \|x\|_1$,

$$\begin{cases} h_i(x,k) = x_i, & \text{for } i < i^* \\ h_i(x,k) = 0, & \text{for } i > i^* \\ h_{i^*}(x,k) = \sum_{i=1}^{i^*} x_i - k, & \text{otherwise,} \end{cases}$$

where

$$i^* = \min\{j | \sum_{i=1}^{j} x_i \geq k\}.$$

Using this operator, we can write the number of jobs remaining in the waitlist at the end of period t as

$$f_t = s_t + \delta_t - h(s_t + \delta_t, d_t + C_t).$$

Next, the waiting cost incurred in period t can be written as

$$W_t = f_t^\tau w.$$

In the next period, the initial state of the system is $s_{t+1} = f_t$.

For each policy Π, we add a superscript Π to all the state and decision variables that result from Π. If Π is an online algorithm, the decision d_t^Π does not depend on any information to be

realized later than period t. When we present our online algorithm in Section 3.2, the objective is the undiscounted total cost over a finite number T of periods, namely, $V_T^{\Pi} = \sum_{t=1}^{T}(d_t^{\Pi} + W_t^{\Pi})$. We will show that our online algorithm gives a total cost which is at most 2 times the total cost under an optimal offline algorithm for any sample path $\{(C_t, \delta_t)\}_t$. In Section 3.3 we further show that the same result holds in discounted, finite and infinite-horizon settings.

3.1. Comparison of Scheduling Policies

In this section we present a method for comparing system states generated by different scheduling policies. Specifically, we construct a *distance function* which, on the one hand, captures the difference in the cumulative overtime usage between two scheduling policies, and on the other hand, indicates whether the jobs under one policy have 'lower' priorities than those under the other policy. We will use the distance function to prove the competitive ratio of our online algorithms.

For two scheduling policies Π and Θ, the distance function $\phi_t(\Pi, \Theta)$ is defined recursively as

$$\begin{cases} \phi_0(\Pi, \Theta) = 0, \\ \phi_t(\Pi, \Theta) = \max\{\phi_{t-1}(\Pi, \Theta) - d_t^{\Pi} + d_t^{\Theta}, 0\} \quad \text{for } t \geq 1, \end{cases} \tag{1}$$

where recall that d_t^{Π} and d_t^{Θ} are the numbers of overtime slots used under Π and Θ in period t, respectively.

Intuitively, the distance function stores the cumulative difference between the number of overtime slots used under Π and Θ. However, its value is always kept non-negative. Table 1 provides an illustrative example. It lists the number of overtime slots used in periods from 1 to 9 on a sample path. The corresponding values of the distance function are shown in the bottom row of the table.

Next, we will show in Theorem 1 that the distance function has a direct physical interpretation. If we remove the number of jobs equal to the value of the function $\phi_t(\Pi, \Theta)$ from state f_t^{Π}, the rest of the jobs in f_t^{Π} will be 'dominated' by the jobs in f_t^{Θ} in terms of priorities. Figure 1 illustrates this interpretation. Before stating Theorem 1, we first formalize the following definition of a dominance relationship and some of its implications.

Table 1 Example of the distance function. Based on the numbers of scheduled overtime slots d_t^Π and d_t^Θ of two scheduling policies Π and Θ, respectively, the values of the distance function $\phi_t(\Pi,\Theta)$ are computed and listed in the bottom row.

Period t	1	2	3	4	5	6	7	8	9
d_t^Θ	0	2	0	0	0	1	0	0	1
d_t^Π	1	0	1	2	0	0	0	1	2
$\phi_t(\Pi,\Theta)$	0	2	1	0	0	1	1	0	0

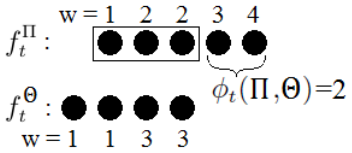

Figure 1 Illustration of the distance function. There are 4 priority classes with waiting costs $w=(4,3,2,1)$. By the end of period t, $f_t^\Pi=(1,1,2,1)$ and $f_t^\Theta=(0,2,0,2)$. The figure displays all the jobs with their waiting costs marked. Assume that $\phi_t(\Pi,\Theta)=2$. After $\phi_t(\Pi,\Theta)=2$ jobs with the highest priorities are removed from f_t^Π, the remaining jobs, marked by the black box, have lower priorities than the jobs in f_t^Θ. Note that if we only removed 1 job with unit waiting cost of 4 from f_t^Π, the remaining jobs would not be 'dominated' by the jobs in f_t^Θ, as there would be 3 jobs with waiting costs of at least 2 remaining in f_t^Π, but only 2 such jobs in f_t^Θ.

DEFINITION 2. For two vectors $x,x'\in\mathbb{Z}_+^n$, we say x is dominated by x' and write $x\preceq x'$ if

$$\sum_{i=1}^{l} x_i \le \sum_{i=1}^{l} x_i' \quad \forall l=1,2,..,n.$$

A result immediately following this definition is that, if x and x' represent different system states at the end of period t, and $x\preceq x'$, then the total waiting cost incurred by the jobs in x at t is no greater than that incurred by the jobs in x'.

LEMMA 1. *For two vectors $x,x'\in\mathbb{Z}_+^n$, if $x\preceq x'$, then $x^\tau w\le x'^\tau w$.*

Proof.

$$\sum_{i=1}^{l} x_i \le \sum_{i=1}^{l} x_i' \quad \forall l=1,2,...,n$$

$$\begin{aligned}
&\Longrightarrow \sum_{i=1}^{l} x_i\alpha \le \sum_{i=1}^{l} x'_i\alpha \quad \forall l = 1,2,...,n \text{ and } \forall \alpha > 0 \\
&\Longrightarrow \sum_{l=1}^{n-1}\sum_{i=1}^{l} x_i(w_l - w_{l+1}) + \sum_{i=1}^{n} x_i w_n \le \sum_{l=1}^{n-1}\sum_{i=1}^{l} x'_i(w_l - w_{l+1}) + \sum_{i=1}^{n} x'_i w_n \\
&\Longrightarrow \sum_{i=1}^{n} x_i w_i \le \sum_{i=1}^{n} x'_i w_i.
\end{aligned}$$

□

The following lemma states two simple operations that preserve a dominance relation:

LEMMA 2. *Fix an integer $l \ge 0$ and two vectors $x, x' \in \mathbb{Z}_+^n$ satisfying*

$$x - h(x,l) \preceq x'.$$

1. *For any integer $l' \ge 0$,*

$$x - h(x, l+l') \preceq x' - h(x', l').$$

2. *For any vector $\delta \in \mathbb{N}_0^n$,*

$$x + \delta - h(x+\delta, l) \preceq x' + \delta.$$

Proof. This lemma is easily proved by directly checking the definition of dominance relation. □

Next, we prove an invariance between two scheduling policies that can be stated in terms of the distance function and the dominance relation. This invariance will help us later to compare the cost of the policies.

THEOREM 1. *For any two scheduling policies Π and Θ, we have*

$$f_t^{\Pi} - h(f_t^{\Pi}, \phi_t(\Pi,\Theta)) \preceq f_t^{\Theta}, \ \forall t = 1,2,...,T. \tag{2}$$

Proof. Recall that $s_t = f_{t-1}$, so equation (2) is equivalent to

$$s_t^{\Pi} - h(s_t^{\Pi}, \phi_{t-1}(\Pi,\Theta)) \preceq s_t^{\Theta}. \tag{3}$$

This equation (3) is clearly true for $t=1$, as $s_1^\Pi = s_1^\Theta$ is the initial state.

Suppose that (3) holds up to period t, we next prove that it is also true for period $t+1$.

In period t, after δ_t new jobs arrive, according to Lemma 2 we have

$$s_t^\Pi + \delta_t - h(s_t^\Pi + \delta_t, \phi_{t-1}(\Pi,\Theta)) \preceq s_t^\Theta + \delta_t.$$

Then after Θ removes $C_t + d_t^\Theta$ jobs, Lemma 2 gives us

$$s_t^\Pi + \delta_t - h(s_t^\Pi + \delta_t, \phi_{t-1}(\Pi,\Theta) + C_t + d_t^\Theta) \preceq s_t^\Theta + \delta_t - h(s_t^\Theta + \delta_t, C_t + d_t^\Theta).$$

Now we let $l = \phi_{t-1}(\Pi,\Theta) + d_t^\Theta - d_t^\Pi$ and rewrite the above equation as

$$s_t^\Pi + \delta_t - h(s_t^\Pi + \delta_t, l + C_t + d_t^\Pi) \preceq f_t^\Theta.$$

Depending on the value of l, there are two cases:

1. If $l < 0$, we have

$$s_t^\Pi + \delta_t - h(s_t^\Pi + \delta_t, C_t + d_t^\Pi) \preceq s_t^\Pi + \delta_t - h(s_t^\Pi + \delta_t, l + C_t + d_t^\Pi)$$

because the left hand side has more jobs removed from the vector $s_t^\Theta + \delta_t$. It is easy to check that the binary relation $\preceq$ is transitive, so the above equation leads to

$$s_t^\Pi + \delta_t - h(s_t^\Pi + \delta_t, C_t + d_t^\Pi) \preceq f_t^\Theta$$

$$\Longrightarrow f_t^\Pi \preceq f_t^\Theta$$

$$\Longrightarrow f_t^\Pi - h(f_t^\Pi, 0) \preceq f_t^\Theta.$$

2. If $l \geq 0$, we have

$$\begin{aligned} s_t^\Pi + \delta_t - h(s_t^\Pi + \delta_t, l + C_t + d_t^\Pi) &= s_t^\Pi + \delta_t - h(s_t^\Pi + \delta_t, C_t + d_t^\Pi) \\ &\quad - h(s_t^\Pi + \delta_t - h(s_t^\Pi + \delta_t, C_t + d_t^\Pi), l) \\ &= f_t^\Pi - h(f_t^\Pi, l) \end{aligned}$$

$$\Longrightarrow f_t^\Pi - h(f_t^\Pi, l) \preceq f_t^\Theta.$$

In sum, we have

$$f_t^\Pi - h(f_t^\Pi, \max(l,0)) \preceq f_t^\Theta$$

$$\Longrightarrow f_t^\Pi - h(f_t^\Pi, \phi_t(\Pi,\Theta)) \preceq f_t^\Theta$$

$$\Longrightarrow s_{t+1}^\Pi - h(s_{t+1}^\Pi, \phi_t(\Pi,\Theta)) \preceq s_{t+1}^\Theta.$$

Thus the theorem is proved. □

The next theorem shows that the distance function separates all periods into two types, depending on the sign of the distance function. In one case, the current waiting cost incurred under policy Π is bounded by that under Θ. In the other case, the cumulative overtime cost incurred under Π is bounded by that under Θ.

THEOREM 2. *In any period t,*

1. *if $\phi_t(\Pi,\Theta) = 0$, then $W_t^\Pi \le W_t^\Theta$;*
2. *if $\phi_t(\Pi,\Theta) > 0$, let $t_0 = \max\{k : \phi_k(\Pi,\Theta) = 0, k < t\}$. Then*

$$\sum_{k=t_0+1}^{t} d_k^\Pi < \sum_{k=t_0+1}^{t} d_k^\Theta.$$

Proof. If $\phi_t(\Pi,\Theta) = 0$, we know from Theorem 1 that $f_t^\Pi \le f_t^\Theta$. Then Lemma 1 gives $W_t^\Pi \le W_t^\Theta$. The case $\phi_t(\Pi,\Theta) > 0$ can be proved by directly checking the definition of the distance function. □

Using Table 1, we illustrate the two types of periods distinguished in Theorem 2.

1. Periods in which $\phi_t(\Pi,\Theta) = 0$. These are periods $t = 1,4,5,8,9$ in Table 1. From the first statement of Theorem 2 we know that for this type of periods, the waiting costs under Π is bounded by the waiting costs under Θ.

2. Periods in which $\phi_t(\Pi,\Theta) > 0$. We can divide these periods into intervals of consecutive periods, e.g., interval $[2,3]$ and interval $[6,7]$ in Table 1. During each of these intervals, the total number of overtime slots used in Π is no greater than the number of overtime slots used in Θ. Hence, in these intervals the total overtime cost under Π is bounded by that under Θ.

In sum, in any type of periods, one cost component of Π, either the waiting cost or the overtime cost, is bounded by the corresponding cost of Θ. Since Θ can be any scheduling policy including the optimal offline policy, Π will have a competitive ratio of 2 if it can balance the two cost components evenly in an online manner. We next show that a simple method of balancing costs leads to a 2-competitive algorithm.

3.2. Online Algorithm

In this section we present a 2-competitive online algorithm for the allocation-scheduling problem without cancellations.

Define an online algorithm OLN as follows. In each period t, OLN balances the total cumulative waiting cost and total cumulative overtime cost by minimizing the maximum of the two. Mathematically, let $W(d) = (s_t^{\text{OLN}} + \delta_t - h(s_t^{\text{OLN}} + \delta_t, d + C_t))^\tau w$ be the waiting cost to be incurred in period t if d overtime slots are used in t. Then d_t^{OLN} is determined by (recall that the unit overtime cost is $p = 1$)

$$d_t^{\text{OLN}} = \arg\min_d \max(\sum_{i=1}^{t-1} d_i^{\text{OLN}} + d, \sum_{i=1}^{t-1} W_i^{\text{OLN}} + W(d)). \tag{4}$$

The idea of OLN is to keep these two cumulative costs as closely matched to each other as possible.

THEOREM 3. *For any policy* Π *and any sample path,*

$$\max(\sum_{i=1}^{t} d_i^{OLN}, \sum_{i=1}^{t} W_i^{OLN}) \le \sum_{i=1}^{t} (d_i^{\Pi} + W_i^{\Pi}), \ \forall t = 1, 2, ..., T. \tag{5}$$

Proof. When $t = 0$ the condition (5) is trivially true. Suppose that (5) is true up to period $t-1$. We next prove that it also holds for period t.

Let $g_t = \max(\sum_{i=1}^{t} d_i^{\text{OLN}}, \sum_{i=1}^{t} W_i^{\text{OLN}})$ be the maximum of the two cumulative costs up to period t.

- Case 1: $\phi_{t-1}(\text{OLN}, \Pi) + d_t^{\Pi} - d_t^{\text{OLN}} < 0$. We immediately have $d_t^{\text{OLN}} > 0$ and

$$\phi_{t-1}(\text{OLN}, \Pi) + d_t^{\Pi} - (d_t^{\text{OLN}} - 1) \le 0.$$

Then from Theorem 2 we know that

$$W(d_t^{\text{OLN}} - 1) \le W_t^{\Pi}.$$

In other words, even if we schedule one fewer job in period t under OLN, the resulting waiting cost for this period is still less than or equal to W_t^{Π}. The decision criterion for OLN in (4) gives us

$$g_t \le g_{t-1} + W(d_t^{\text{OLN}} - 1)$$

because otherwise using $d_t^{\text{OLN}} - 1$ overtime slots instead of d_t^{OLN} in period t would reduce the maximum component of cumulative costs. Connecting the above two equations we get

$$g_t \le g_{t-1} + W(d_t^{\text{OLN}} - 1) \le g_{t-1} + W_t^{\Pi} \le \sum_{i=1}^{t-1}(d_i^{\Pi} + W_i^{\Pi}) + W_t^{\Pi} \le \sum_{i=1}^{t}(d_i^{\Pi} + W_i^{\Pi}),$$

where the third inequality follows from induction on the $(t-1)$-th period.

- Case 2: $\phi_{t-1}(\text{OLN}, \Pi) + d_t^{\Pi} - d_t^{\text{OLN}} > 0$. Again let

$$t_0 = \max\{k : \phi_k(\text{OLN}, \Pi) = 0, k < t\} \tag{6}$$

be the last period in which the distance function was equal to 0. Since in this case $\phi_t(\text{OLN}, \Pi) = \phi_{t-1}(\text{OLN}, \Pi) + d_t^{\Pi} - d_t^{\text{OLN}} > 0$, from Theorem 2 we know that

$$\sum_{i=t_0+1}^{t} d_i^{\text{OLN}} < \sum_{i=t_0+1}^{t} d_i^{\Pi}$$
$$\Longrightarrow \sum_{i=t_0+1}^{t} d_i^{\text{OLN}} + 1 \le \sum_{i=t_0+1}^{t} d_i^{\Pi}.$$

On the other hand, definition (4) gives us

$$g_t \le g_{t_0} + (\sum_{i=t_0+1}^{t} d_i^{\text{OLN}} + 1)$$

because otherwise we could use one more overtime slot to reduce g_t. Combining the above two equations we get

$$g_t \le g_{t_0} + (\sum_{i=t_0+1}^{t} d_i^{\text{OLN}} + 1) \le g_{t_0} + \sum_{i=t_0+1}^{t} d_i^{\Pi} \le \sum_{i=1}^{t_0}(d_i^{\Pi} + W_i^{\Pi}) + \sum_{i=t_0+1}^{t} d_i^{\Pi} \le \sum_{i=1}^{t}(d_i^{\Pi} + W_i^{\Pi}),$$

where the third inequality comes from induction on the t_0-th period.

- Case 3a: $\phi_{t-1}(\text{OLN}, \Pi) + d_t^{\Pi} - d_t^{\text{OLN}} = 0$, $g_t = \sum_{i=1}^{t} d_i^{\text{OLN}}$. Let t_0 be defined as in (6). From the definition of the distance function we know that

$$\sum_{i=t_0+1}^{t} d_i^{\text{OLN}} = \sum_{i=t_0+1}^{t} d_i^{\Pi}.$$

Then we have

$$g_t \le g_{t_0} + \sum_{i=t_0+1}^{t} d_i^{\text{OLN}} \le \sum_{i=1}^{t_0} (d_i^{\Pi} + W_i^{\Pi}) + \sum_{i=t_0+1}^{t} d_i^{\Pi} \le \sum_{i=1}^{t} (d_i^{\Pi} + W_i^{\Pi}),$$

where the first inequality comes from the condition for this case, namely that $g_t = \sum_{i=1}^{t} d_i^{\text{OLN}}$.

- Case 3b: $\phi_{t-1}(\text{OLN}, \Pi) + d_t^{\Pi} - d_t^{\text{OLN}} = 0$, $g_t = \sum_{i=1}^{t} W_i^{\text{OLN}}$. From Theorem 2 we have

$$W_t^{\text{OLN}} \le W_t^{\Pi}$$

$$\implies g_t \le g_{t-1} + W_t^{\text{OLN}} \le g_{t-1} + W_t^{\Pi} \le \sum_{i=1}^{t-1} (d_i^{\Pi} + W_i^{\Pi}) + W_t^{\Pi} \le \sum_{i=1}^{t} (d_i^{\Pi} + W_i^{\Pi}),$$

where the first inequality comes from the condition for this case, namely that $g_t = \sum_{i=1}^{t} W_i^{\text{OLN}}$.

□

Finally using Theorem 3 we can show that OLN is 2-competitive, by letting Π be the optimal offline algorithm OFF.

COROLLARY 1. *On every sample path,*

$$\sum_{i=1}^{T} (d_i^{OLN} + W_i^{OLN}) \le 2 \sum_{i=1}^{T} (d_i^{OFF} + W_i^{OFF}).$$

Proof.

$$\sum_{i=1}^{T} (d_i^{\text{OLN}} + W_i^{\text{OLN}}) \le 2 \max(\sum_{i=1}^{T} d_i^{\text{OLN}}, \sum_{i=1}^{T} W_i^{\text{OLN}}) \le 2 \sum_{i=1}^{T} (d_i^{\text{OFF}} + W_i^{\text{OFF}}).$$

□

3.3. Generalization to Discounted Costs

Now we generalize our previous results to the case of discounted future costs. Given a discount factor $\gamma \in (0,1)$, let the total discounted cost from period 1 to T be $V_T^{\Pi}(\gamma)$,

$$V_T^{\Pi}(\gamma) = \sum_{t=1}^{T} (d_t^{\Pi} p + \phi_t^{\Pi}) \gamma^{t-1}.$$

The following theorem ensures that the competitive ratio of our online algorithm is still 2 in the discounted-cost case.

THEOREM 4. *For any policy* Π *and any horizon* T*, where* T *is possibly infinite, we have*

$$V_T^{OLN}(\gamma) \leq 2V_T^{\Pi}(\gamma).$$

Proof. We already know from Corollary 1 that for any length t of the horizon and any sample path we have

$$V_t^{\mathrm{OLN}} \leq 2V_t^{\Pi},$$

where V_t^{Π} is the undiscounted cost from periods 1 to t. Then for any policy Π,

$$\begin{aligned}
V_T^{\mathrm{OLN}}(\gamma) &= \sum_{t=1}^{T} (d_t^{\mathrm{OLN}} p + \phi_t^{\mathrm{OLN}}) \gamma^{t-1} \\
&= \sum_{t=1}^{T} (V_t^{\mathrm{OLN}} - V_{t-1}^{\mathrm{OLN}}) \gamma^{t-1} \\
&= \sum_{t=1}^{T-1} V_t^{\mathrm{OLN}} \cdot (\gamma^{t-1} - \gamma^t) + V_T^{\mathrm{OLN}} \cdot \gamma^{T-1} \\
&\leq \sum_{t=1}^{T-1} 2V_t^{\Pi} \cdot (\gamma^{t-1} - \gamma^t) + 2V_T^{\Pi} \cdot \gamma^{T-1} \\
&= 2V_T^{\Pi}(\gamma).
\end{aligned}$$

□

3.4. Lower Bounds

We prove that our online algorithm achieves the optimal competitive ratio by reducing our scheduling problem into a *ski-rental problem*, and concluding that the competitive ratios for the ski-rental problem apply to our model.

The classical ski-rental problem, which is first studied by Karlin et al. (1988), is a simplified version of our allocation-scheduling problem. In the ski-rental problem, a single job waits to be processed some time in the future, but the exact date that the job will be processed is unknown. A waiting cost of \$1 is incurred in each period that the job has to wait. The job can also be immediately processed at an additional cost of $\$B$ at any time. If we know that the job has to wait at least B periods, then it is optimal to immediately process the job in the current period. If the job needs to wait no more than B periods, then it is optimal to let it wait. This ski-rental problem is online if the exact time that the job will be processed is unknown and is chosen by an adversary. It is well known that the optimal competitive ratio of the ski-rental problem is 2 for deterministic algorithms (Karlin et al. 1988) and $e/(e-1)$ for randomized algorithms (Karlin et al. 1990).

THEOREM 5. *OLN is an optimal online algorithm for the allocation-scheduling model.*

Proof. In our allocation scheduling model, if there is only one job in the system and we always let $C_t = 0$ until some future period chosen by an adversary, then the problem reduces to the ski-rental problem. Thus, the ski-rental problem is a subclass of the allocation-scheduling problem. Therefore, its lower bounds on the competitive ratio also apply to the algorithms for the allocation-scheduling problem. From this, we can conclude that our 2-competitive deterministic algorithm has the lowest possible competitive ratio. □

4. Model of Allocation Scheduling with Cancellations

In this section, we consider the allocation-scheduling problem with cancellations. The online algorithm we propose in this section is adapted from the cost-balancing algorithm of the previous section. The algorithm in this section is a deterministic one. We will only prove the competitive ratio over an undiscounted and finite horizon, but similar to the results in Section 3.3, our competitive analysis can be easily generalized to a discounted and infinite horizon.

Starting from the model without cancellations, we assume that a class i job has a cancellation probability of $q_i \in [0,1]$, and a cancellation cost of $r_i \geq 0$. We assume that the cancellation cost dominates the overtime cost for each class, i.e., $r_i \geq p = 1$ for all i. We further assume that the

cancellation probabilities and costs are higher for higher-priority classes, i.e., $r_1 \geq r_2 \geq \cdots \geq r_n$, and $q_1 \geq q_2 \geq \cdots \geq q_n$. This assumption makes sense in most applications. In healthcare, higher priority patients have a higher need to be seen quickly, less willingness to wait, and higher tendency to leave for other care arrangements if they are made to wait for too long. In server applications, higher priority jobs have shorter deadlines. In service systems, higher priority customers are more impatient to wait, and often bring higher profits to the system which would be lost if they leave the queue.

In each period, the following events happen in sequence

1. At the beginning of period t, $s_t = (s_{t1}, s_{t2}, ..., s_{tn})$ is the total number of jobs in system, where s_{ti} is the number of jobs of class i.

2. Each job in class i independently leaves the system with probability q_i. The remaining jobs form a state m_t, $m_t \leq s_t$. The total cancellation cost incurred in period t is

$$R_t = (s_t - m_t)^\tau r.$$

3. The capacity C_t and new arrivals δ_t are observed. The system state becomes $m_t + \delta_t$.

4. The scheduling decision d_t for period t is made. The number of jobs remaining in the queue is f_t, $f_t = m_t + \delta_t - h(m_t + \delta_t, d_t + C_t)$. The overtime cost incurred in period t is d_t, and the waiting cost incurred is

$$W_t = f_t^\tau w.$$

5. In the next period we have $s_{t+1} = f_t$.

For the competitive analysis of the online algorithm with cancellations, we assume that an offline algorithm sees future arrivals and capacities, δ_t, C_t, $t = 1, 2, ..., T$, but does not see which jobs will cancel. Let $\mathcal{F} = \sigma(\delta_1, \delta_2, ..., \delta_T, C_1, C_2, ..., C_T)$ contain the information that an offline algorithm can see. The objective is

$$\mathbf{E}[V_T|\mathcal{F}] = \mathbf{E}[\sum_{t=1}^{T}(R_t + d_t + W_t)|\mathcal{F}],$$

where the expectation is taken over the random cancellation events.

Before presenting the online algorithm, it is necessary to reexamine the question of, in the presence of job cancellations, whether it is still optimal for the offline algorithm to serve jobs with the highest priorities first, i.e., whether we can still use the $h(\cdot,\cdot)$ operator to represent an optimal offline scheduling decision. The following theorem ensures that this service rule is still optimal.

THEOREM 6. *The optimal offline algorithm OFF always schedules jobs with the highest priorities in each period.*

Proof. As an offline algorithm, OFF knows all the arrivals and capacities upfront. However, since the cancellation events are exogenous to offline algorithms, OFF faces a stochastic setting in which jobs cancel randomly in each period. In this stochastic decision process, let $u_t^1(s)$ be the expected cost of OFF from t to T when the system state at t is s. That is, let

$$u_t^1(s) = \mathbf{E}[\sum_{i=t}^{T}(R_i^{\text{OFF}} + d_i^{\text{OFF}} + W_i^{\text{OFF}})|\mathcal{F}, s_t = s].$$

Let $u_t^2(s)$ be the cost of OFF from t to T immediately after cancellations have occurred in period t, and when the system state at t is s,

$$u_t^2(s) = \mathbf{E}[d_t^{\text{OFF}} + W_t^{\text{OFF}} + \sum_{i=t+1}^{T}(R_i^{\text{OFF}} + d_i^{\text{OFF}} + W_i^{\text{OFF}})|\mathcal{F}, m_t = s].$$

We next show by induction that for any $s_1 \preceq s_2$,

$$u_t^1(s_1) \leq u_t^1(s_2), \text{ and} \tag{7}$$

$$u_t^2(s_1) \leq u_t^2(s_2). \tag{8}$$

These two results will naturally lead to the proof of this theorem.

First, it is clear that (8) holds in the last period T, as no cancellation will ever happen starting at that time, and hence the result reduces to the case without cancellations. Suppose that (8) holds starting from period t. We next prove that (7) also holds for period t and that (8) holds for period $t-1$.

Let e_i be the unit vector with 1 for the ith element and 0 for all other elements. Since adding more jobs to the system only imposes a larger cost, we must have

$$u_t^1(s) \le u_t^1(s+e_i)$$

for any $i=1,2,...,n$. Then to prove (7) it suffices to prove that for any $i>j$,

$$u_t^1(s+e_j) \le u_t^1(s+e_i).$$

For any $\tilde{s} \le s$, let $P(s,\tilde{s})$ be the probability that all the jobs in $\tilde{s}$ remain while all the jobs in $s-\tilde{s}$ cancel. Then the offline cost value can be written as

$$u_t^1(s+e_i) = \sum_{\tilde{s}\le s} P(s,\tilde{s}) \left[(s-\tilde{s})^\tau r + q_i(r_i + u_t^2(\tilde{s})) + (1-q_i)u_t^2(\tilde{s}+e_i)\right],$$

where $r_i + u_t^2(\tilde{s})$ is the total cost value under the condition that the additional job e_i cancels, and $u_t^2(\tilde{s}+e_i)$ is the cost value under the condition that the additional job does not cancel.

By induction we know that $u_t^2(\tilde{s}+e_j) \le u_t^2(\tilde{s}+e_i)$ if $i>j$. Moreover, the marginal cost of $u_t^2(\cdot)$ must be bounded by the overtime cost, namely,

$$u_t^2(\tilde{s}+e_i) - u_t^2(\tilde{s}) \le p \le r_i$$

because otherwise the offline policy would service the additional job e_i by overtime and reduce the marginal cost to p. Then for any $i>j$,

$$\begin{aligned}
u_t^1(s+e_j) &= \sum_{\tilde{s}\le s} P(s,\tilde{s}) \left[(s-\tilde{s})^\tau r + q_j(r_j + u_t^2(\tilde{s})) + (1-q_j)u_t^2(\tilde{s}+e_j)\right] \\
&\le \sum_{\tilde{s}\le s} P(s,\tilde{s}) \left[(s-\tilde{s})^\tau r + q_j(r_i + u_t^2(\tilde{s})) + (1-q_j)u_t^2(\tilde{s}+e_i)\right] \\
&\le \sum_{\tilde{s}\le s} P(s,\tilde{s}) \left[(s-\tilde{s})^\tau r + q_i(r_i + u_t^2(\tilde{s})) + (1-q_i)u_t^2(\tilde{s}+e_i)\right] \\
&= u_t^1(s+e_i).
\end{aligned}$$

where the last inequality follows from the fact that $q_i > q_j$ and that $r_i + u_t^2(\tilde{s}) \ge u_t^2(\tilde{s}+e_i)$.

Thus we have proved (7) for period t. Now it is immediately clear that the optimal offline scheduling rule in period $t-1$ always services the jobs with the highest priorities, because (7) states that it is better to have lower priority jobs in the system at the beginning of period t, and that the costs to serve any two jobs are the same. It also follows that (8) holds for period $t-1$, as having lower-priority jobs in system leads to lower waiting costs and, at the same time, lower-priority jobs at the beginning of the next period. $\square$

4.1. Coupling of two scheduling policies

Again we want to compare the system states under two policies Π and Θ. Suppose they start with the same initial state and experience the same capacity C_t and arrivals δ_t for each period t. Since both online and offline algorithms do not know which jobs will cancel, we can couple the cancellation events under Π and Θ. We show that a new distance function can be defined based on the coupling of cancellations.

Let $o_t^\Pi = \|s_t^\Pi - m_t^\Pi\|_1$ be the total number of cancelled jobs in period t for policy Π. We define a new distance function $\bar{\phi}(\Pi,\Theta)$ for any two policies Π and Θ as follows

$$\begin{cases} \bar{\phi}_0(\Pi,\Theta) = 0, \\ \bar{\phi}_t(\Pi,\Theta) = \max\{\bar{\phi}_{t-1}(\Pi,\Theta) - d_t^\Pi - o_t^\Pi + d_t^\Theta + o_t^\Theta, 0\} \quad \text{for } t \geq 1. \end{cases} \tag{9}$$

This new distance function takes both the number of overtime slots and the number of cancelled jobs into account.

Suppose that at the beginning of period t we have

$$s_t^\Pi - h(s_t^\Pi, \phi_{t-1}(\Pi,\Theta)) \preceq s_t^\Theta. \tag{10}$$

Then we can always simulate the cancellations in period t in three phases as follows (see Figure 2 for an illustration):

1. Let the $\bar{\phi}_{t-1}(\Pi,\Theta)$ jobs with the highest priorities in state s_t^Π, i.e., those counted in $h(s_t^\Pi, \bar{\phi}_{t-1}(\Pi,\Theta))$, make their cancellation decisions.

2. Let $l = (\|s_t^\Pi\|_1 - \bar{\phi}_{t-1}(\Pi, \Theta))^+$ be the number of remaining jobs in state s_t^Π that have not made their cancellation decisions yet. Let $U_1, U_2, ..., U_l$ be i.i.d. $[0,1]$ uniform random variables. For each of the l jobs, going from the highest priority to the lowest priority, if the ith job is in class j, let the ith job cancel if and only if $q_j \geq U_i$. Then, for the l jobs with the highest priorities in state s_t^Θ, let them cancel similarly, by using the same sequence of uniform random variables $U_1, U_2, ..., U_l$ (but using possibly different cancellation probabilities). In this way we have coupled the cancellation events between the l jobs with the lowest priorities under Π and the l jobs with the highest priorities under Θ.

3. Let the other jobs in s_t^Θ make their cancellation decisions.

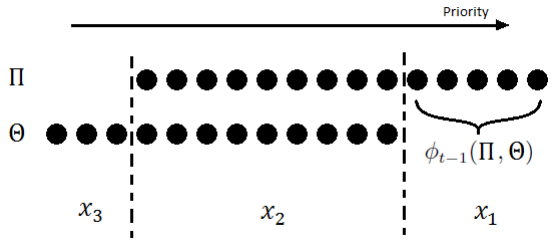

Figure 2 Stochastic coupling of cancellation events. After removing $\phi_{t-1}(\Pi, \Theta)$ jobs with the highest priorities from the state under Π, the remaining l jobs are dominated by the jobs under Θ, in that the priority of each remaining job under Π is at most that of the job with the same priority ranking under Θ. In Phase 2, the cancellation events of each pair of jobs having the same priority ranking under the two policies are coupled together.

The following theorem shows that, under the above coupling of cancellation events, the distance function still enables us to set up a dominance relationship between Π and Θ.

THEOREM 7. *Suppose (10) holds in period t. After the above coupled cancellation process, we have on every sample path,*

$$m_t^\Pi - h(m_t^\Pi, \bar{\phi}_{t-1}(\Pi, \Theta) - o_t^\Pi + o_t^\Theta) \preceq m_t^\Theta. \tag{11}$$

In particular,

$$\bar{\phi}_{t-1}(\Pi, \Theta) - o_t^\Pi + o_t^\Theta \geq 0. \tag{12}$$

By the end of period t,

$$f_t^\Pi - h(f_t^\Pi, \bar{\phi}_{t-1}(\Pi,\Theta) - d_t^\Pi - o_t^\Pi + d_t^\Theta + o_t^\Theta) \preceq f_t^\Theta. \tag{13}$$

Proof. Recall that $l = (\|s_t^\Pi\|_1 - \bar{\phi}_{t-1}(\Pi,\Theta))^+$. Let x_i^Π and x_i^Θ be the vectors of jobs considered in the ith coupling phase under Π and Θ, respectively, for $i = 1,2,3$ (see Figure 2). In particular, we have in Phase 1,

$$x_1^\Pi = h(s_t^\Pi, \bar{\phi}_{t-1}(\Pi,\Theta)),$$

in Phase 2,

$$x_2^\Pi = s_t^\Pi - x_1^\Pi \quad \text{and} \quad x_2^\Theta = h(s_t^\Theta, l),$$

and in Phase 3,

$$x_3^\Theta = s_t^\Theta - x_2^\Theta.$$

Let $\bar{x}_i^\Pi$ and $\bar{x}_i^\Theta$ be the vectors of remaining jobs in x_i^Π and x_i^Θ, respectively, after cancellations have occurred. Let $y_i^\Pi = \|x_i^\Pi\|_1 - \|\bar{x}_i^\Pi\|_1$ and $y_i^\Theta = \|x_i^\Theta\|_1 - \|\bar{x}_i^\Theta\|_1$ be the number of cancelled jobs in phase i under Π and Θ, respectively.

Under coupling, the ith job in x_2^Π, ranked by priority, is coupled with the ith job in x_2^Θ. According to the initial condition (10), we have $x_2^\Pi \preceq x_2^\Theta$, i.e., the ith job in x_2^Π has equal or lower priority than the ith job in x_2^Θ. According to the coupling process, if the ith job in x_2^Π cancels, then the ith job in x_2^Θ cancels. So we must have

$$y_2^\Pi \le y_2^\Theta.$$

Since $x_2^\Pi \preceq x_2^\Theta$, by removing $y_2^\Theta - y_2^\Pi$ jobs with the highest priorities from $\bar{x}_2^\Pi$, the resulting state must be dominated by $\bar{x}_2^\Theta$, i.e.,

$$\bar{x}_2^\Pi - h(\bar{x}_2^\Pi, y_2^\Theta - y_2^\Pi) \preceq \bar{x}_2^\Theta.$$

In phase 1, there are $\bar{\phi}_{t-1}(\Pi,\Theta) - y_1^\Pi$ jobs in $\bar{x}_1^\Pi$. Plugging these jobs into the dominance relation, we get

$$\bar{x}_1^\Pi + \bar{x}_2^\Pi - h(\bar{x}_1^\Pi + \bar{x}_2^\Pi, \bar{\phi}_{t-1}(\Pi,\Theta) - y_1^\Pi + y_2^\Theta - y_2^\Pi) \preceq \bar{x}_2^\Theta.$$

By further adding the jobs in phase 3, we get

$$\bar{x}_1^\Pi + \bar{x}_2^\Pi - h(\bar{x}_1^\Pi + \bar{x}_2^\Pi, \bar{\phi}_{t-1}(\Pi,\Theta) - y_1^\Pi + y_2^\Theta - y_2^\Pi + y_3^\Theta) \preceq \bar{x}_2^\Theta + x_3^\Theta,$$

which is just (11).

To prove (12), note that $\bar{\phi}_{t-1}(\Pi,\Theta) \geq y_1^\Pi$ because no more than $\bar{\phi}_{t-1}(\Pi,\Theta)$ jobs can cancel in phase 1, and $y_2^\Theta \geq y_2^\Pi$ due to the coupling process. Hence

$$\bar{\phi}_{t-1}(\Pi,\Theta) - o_t^\Pi + o_t^\Theta = \bar{\phi}_{t-1}(\Pi,\Theta) - y_1^\Pi + y_2^\Theta - y_2^\Pi + y_3^\Theta \geq \bar{\phi}_{t-1}(\Pi,\Theta) - y_1^\Pi + y_2^\Theta - y_2^\Pi \geq 0,$$

proving (12).

To see that (11) and (12) lead to (13), we treat m_t^Π and m_t^Θ as states in an intermediate period, and treat $\bar{\phi}_{t-1}(\Pi,\Theta) - o_t^\Pi + o_t^\Theta \geq 0$ as the distance function value for the intermediate period. Then (13) follows according to Theorem 1.

□

Corollary 2. *For any two scheduling policies* Π *and* Θ*, we have on every sample path,*

$$f_t^\Pi - h(f_t^\Pi, \bar{\phi}_t(\Pi,\Theta)) \preceq f_t^\Theta, \ \forall t = 1,2,...,T.$$

Proof. This statement is equivalent to equation (13), which is also the same as

$$s_{t+1}^\Pi - h(s_{t+1}^\Pi, \bar{\phi}_t(\Pi,\Theta)) \preceq s_{t+1}^\Theta.$$

This finishes the proof that (10) holds for all period t by induction. Therefore, (13) holds for all period t. □

In the remainder of this paper, we use the coupling of cancellations whenever we compare two scheduling policies. We will directly use Theorem 7 and Corollary 2 without each time specifying that cancellations are coupled.

4.2. New Cost-Accounting Scheme

Next we present a new cost-accounting scheme that separates each cancellation cost into two parts: $r_i - p$ and p (recall that $r_i > p$ for each i). The online algorithm incorporates the two parts of the cancellation cost into the original waiting cost and overtime cost respectively. It achieves a competitive ratio of two by rebalancing the two components.

In the new cost-accounting scheme, let the new cancellation cost be $\tilde{r}_i = p = 1$ for all class i, and let the new waiting cost in period t for class i jobs be

$$\tilde{w}_{t,i} = \begin{cases} w_i + \gamma(r_i - 1)q_i & \text{for period } t < T, \\ w_i & \text{for period } t = T < \infty, \end{cases}$$

where γ is the discount factor. Note that the new waiting cost in the last period is different from that in other periods. It is also easy to check that $\tilde{w}_{t,1} \geq \tilde{w}_{t,2} \geq \cdots \geq \tilde{w}_{t,n}$, and thus $f_t^{\Pi} \preceq f_t^{\Theta}$ implies $\tilde{w}_t^{\intercal} f_t^{\Pi} \leq \tilde{w}_t^{\intercal} f_t^{\Theta}$ for all t.

The idea of the new cost-accounting scheme is that, by setting the new cancellation cost equal to the overtime cost, we can treat a cancellation as a job that is forced to be served in overtime. With this change, we can apply the proof of the performance bound in Theorem 3 with a few changes.

Now the total waiting cost in period t is

$$\tilde{W}_t = f_t^{\intercal} \tilde{w}_t.$$

And the total cost in period t can be written as

$$\tilde{\Omega}_t = (o_t + d_t) + \tilde{W}_t.$$

The following theorem states that the new cost-accounting scheme is equivalent to the original one.

THEOREM 8. *For any horizon T, where T can be infinite, the total cost for any scheduling policy differs only by a constant between the original and new cost-accounting scheme.*

Proof. Let Ω_t and $\tilde{\Omega}_t$ be the cost incurred in period t under the original and new cost-accounting schemes, respectively. We have for any scheduling policy,

$$
\begin{aligned}
&E[\sum_{t=1}^{T}\gamma^{t-1}\Omega_t|\mathcal{F}] \\
&\quad= E[\sum_{t=1}^{T}\gamma^{t-1}((s_t-m_t)^\tau r+d_t+f_t^\tau w)|\mathcal{F}] \\
&\quad= E[\sum_{t=1}^{T}\gamma^{t-1}((s_t-m_t)^\tau (r-\mathbf{1})+\|s_t-m_t\|_1+d_t+f_t^\tau w)|\mathcal{F}] \\
&\quad= E[\sum_{t=1}^{T}\gamma^{t-1}((s_t-m_t)^\tau (r-\mathbf{1})+o_t+d_t+f_t^\tau w)|\mathcal{F}] \\
&\quad= E[\sum_{t=1}^{T}\gamma^{t-1}(E[(s_t-m_t)^\tau (r-\mathbf{1})|\mathcal{F},s_t]+o_t+d_t+f_t^\tau w)|\mathcal{F}] \\
&\quad= E[\sum_{t=1}^{T}\gamma^{t-1}(\sum_{i=1}^{n}s_{ti}q_i(r_i-1)+o_t+d_t+f_t^\tau w)|\mathcal{F}] \\
&\quad= \sum_{i=1}^{n}s_{1,i}q_i(r_i-1)+E[\sum_{t=2}^{T}\gamma^{t-1}\sum_{i=1}^{n}s_{ti}q_i(r_i-1)+\sum_{t=1}^{T}\gamma^{t-1}(o_t+d_t+f_t^\tau w)|\mathcal{F}] \\
&\quad= \sum_{i=1}^{n}s_{1,i}q_i(r_i-1)+E[\sum_{t=1}^{T-1}\gamma^{t-1}\sum_{i=1}^{n}f_{t,i}\gamma q_i(r_i-1)+\sum_{t=1}^{T}\gamma^{t-1}(o_t+d_t+f_t^\tau w)|\mathcal{F}] \\
&\quad= \sum_{i=1}^{n}s_{1,i}q_i(r_i-1)+E[\sum_{t=1}^{T}\gamma^{t-1}(o_t+d_t+f_t^\tau \tilde{w}_t)|\mathcal{F}] \\
&\quad= \sum_{i=1}^{n}s_{1i}q_i(r_i-1)+E[\sum_{t=1}^{T}\gamma^{t-1}\tilde{\Omega}_t|\mathcal{F}].
\end{aligned}
$$

Note that the second term on the last line is the total cost value under the new cost-accounting scheme, and the first term is a constant that depends only on the initial state.

□

This theorem implies that the optimal policy remains the same under the new cost-accounting scheme. Moreover, since the total cost value decreases by a constant $\sum_{i=1}^{n}s_{1i}q_i(r_i-1)$ when new costs are applied, the online algorithms under the new cost-accounting scheme are also online algorithms for the original costs, with the same competitive ratios. We next construct the online algorithms under the new cost-accounting scheme.

4.3. Online Algorithm

In the presence of cancellations, our online algorithm OLN balances the waiting cost $\tilde{W}_t^{\text{OLN}}$ and the sum of overtime cost and cancellation cost by minimizing the maximum of the cumulative cost components. Mathematically, let $W(d) = (m_t^{\text{OLN}} + \delta_t - h(m_t^{\text{OLN}} + \delta_t, d + C_t))^\tau \tilde{w}_t$ be the waiting cost to be incurred in period t if d overtime slots are used in t. Then d_t^{OLN} is determined by

$$d_t^{\text{OLN}} = \arg\min_d \ \max(\sum_{i=1}^{t} o_i^{\text{OLN}} + \sum_{i=1}^{t-1} d_i^{\text{OLN}} + d, \sum_{i=1}^{t-1} \tilde{W}_i^{\text{OLN}} + W(d)). \tag{14}$$

THEOREM 9. *For any policy* Π *and any sample path,*

$$\max(\sum_{i=1}^{t}(o_i^{OLN} + d_i^{OLN}), \sum_{i=1}^{t} \tilde{W}_i^{OLN}) \le \sum_{i=1}^{t}(o_i^{\Pi} + d_i^{\Pi} + \tilde{W}_i^{\Pi}), \ \forall t = 1, 2, ..., T. \tag{15}$$

Proof. The proof is similar to the proof for the deterministic algorithm without cancellations. When $t = 0$ the condition (15) is trivially true. Suppose that (15) is true up to period $t-1$. We next prove that it also holds in period t.

Let $g_t = \max(\sum_{i=1}^{t} o_i^{\text{OLN}} + d_i^{\text{OLN}}), \sum_{i=1}^{t} \tilde{W}_i^{\text{OLN}})$ be the maximum of the two cumulative costs up to period t.

- Case 1: $\bar{\phi}_{t-1}(\text{OLN}, \Pi) + o_t^{\Pi} + d_t^{\Pi} - o_t^{\text{OLN}} - d_t^{\text{OLN}} < 0$. According to Theorem 7 we have $\bar{\phi}_{t-1}(\text{OLN}, \Pi) + o_t^{\Pi} - o_t^{\text{OLN}} \ge 0$. So we must have $d_t^{\text{OLN}} > 0$ and

$$\bar{\phi}_{t-1}(\text{OLN}, \Pi) + o_t^{\Pi} + d_t^{\Pi} - o_t^{\text{OLN}} - (d_t^{\text{OLN}} - 1) \le 0,$$

which means that the distance function in period t will be 0 even if we schedule one fewer overtime slot. Then according to Corollary 2 we know that

$$W(d_t^{\text{OLN}} - 1) \le \tilde{W}_t^{\Pi}.$$

On the other hand, the definition of OLN (14) gives us

$$g_t \le g_{t-1} + W(d_t^{\text{OLN}} - 1)$$

because otherwise using $d_t^{\text{OLN}} - 1$ overtime slots instead of d_t^{OLN} in period t would reduce the maximum component of cumulative costs. Connecting the above two equations we get

$$g_t \le g_{t-1} + W(d_t^{\text{OLN}} - 1) \le g_{t-1} + \tilde{W}_t^{\Pi} \le \sum_{i=1}^{t-1}(o_i^{\Pi} + d_i^{\Pi} + \tilde{W}_i^{\Pi}) + \tilde{W}_t^{\Pi} \le \sum_{i=1}^{t}(o_i^{\Pi} + d_i^{\Pi} + \tilde{W}_i^{\Pi}),$$

where the third inequality follows from induction on the $(t-1)$th period.

- Case 2: $\bar{\phi}_{t-1}(\text{OLN}, \Pi) + o_t^{\Pi} + d_t^{\Pi} - o_t^{\text{OLN}} - d_t^{\text{OLN}} > 0$. Let

$$t_0 = \max\{k : \bar{\phi}_k(\text{OLN}, \Pi) = 0, k < t\} \tag{16}$$

be the last period for which the distance function equals 0. According to the definition of the new distance function we know that

$$\sum_{i=t_0+1}^{t} (o_i^{\text{OLN}} + d_i^{\text{OLN}}) < \sum_{i=t_0+1}^{t} (o_i^{\Pi} + d_i^{\Pi})$$

$$\implies \sum_{i=t_0+1}^{t} (o_i^{\text{OLN}} + d_i^{\text{OLN}}) + 1 \le \sum_{i=t_0+1}^{t} (o_i^{\Pi} + d_i^{\Pi}).$$

On the other hand, definition (14) gives us

$$g_t \le g_{t_0} + \sum_{i=t_0+1}^{t} (o_i^{\text{OLN}} + d_i^{\text{OLN}}) + 1$$

because otherwise we could use one more overtime slot to reduce g_t. Combining the above two equations we get

$$\begin{aligned} g_t \le & g_{t_0} + \sum_{i=t_0+1}^{t} (o_i^{\text{OLN}} + d_i^{\text{OLN}}) + 1 \\ \le & g_{t_0} + \sum_{i=t_0+1}^{t} (o_i^{\Pi} + d_i^{\Pi}) \\ \le & \sum_{i=1}^{t_0}(o_i^{\Pi} + d_i^{\Pi} + \tilde{W}_i^{\Pi}) + \sum_{i=t_0+1}^{t} (o_i^{\Pi} + d_i^{\Pi}) \\ \le & \sum_{i=1}^{t}(o_i^{\Pi} + d_i^{\Pi} + \tilde{W}_i^{\Pi}), \end{aligned}$$

where the third inequality comes from induction on the t_0-th period.

- Case 3a: $\bar{\phi}_{t-1}(\text{OLN}, \Pi) + o_t^\Pi + d_t^\Pi - o_t^{\text{OLN}} - d_t^{\text{OLN}} = 0$, $g_t = \sum_{i=1}^t (o_i^{\text{OLN}} + d_i^{\text{OLN}})$. Let t_0 be defined as in (16). From the definition of the distance function we know that

$$\sum_{i=t_0+1}^{t} (o_i^{\text{OLN}} + d_i^{\text{OLN}}) = \sum_{i=t_0+1}^{t} (o_i^\Pi + d_i^\Pi).$$

Then we have

$$g_t \le g_{t_0} + \sum_{i=t_0+1}^{t} (o_i^{\text{OLN}} + d_i^{\text{OLN}}) \le \sum_{i=1}^{t_0} (o_i^\Pi + d_i^\Pi + \tilde{W}_i^\Pi) + \sum_{i=t_0+1}^{t} (o_i^\Pi + d_i^\Pi) \le \sum_{i=1}^{t} (o_i^\Pi + d_i^\Pi + \tilde{W}_i^\Pi),$$

where the first inequality comes from the condition of this case, namely that $g_t = \sum_{i=1}^t (o_i^{\text{OLN}} + d_i^{\text{OLN}})$.

- Case 3b: $\bar{\phi}_{t-1}(\text{OLN}, \Pi) + o_t^\Pi + d_t^\Pi - o_t^{\text{OLN}} - d_t^{\text{OLN}} = 0$, $g_t = \sum_{i=1}^t \tilde{W}_i^{\text{OLN}}$. From Corollary 2 we have

$$f_t^{\text{OLN}} \preceq f_t^\Pi$$

$$\implies \tilde{W}_t^{\text{OLN}} \le \tilde{W}_t^\Pi$$

$$\implies g_t \le g_{t-1} + \tilde{W}_t^{\text{OLN}} \le g_{t-1} + \tilde{W}_t^\Pi \le \sum_{i=1}^{t-1} (o_i^\Pi + d_i^\Pi + \tilde{W}_i^\Pi) + \tilde{W}_t^\Pi \le \sum_{i=1}^{t} (o_i^\Pi + d_i^\Pi + \tilde{W}_i^\Pi),$$

where the first inequality comes from the condition of this case, namely that $g_t = \sum_{i=1}^t \tilde{W}_i^{\text{OLN}}$.

□

Finally using Theorem 9 we can show that OLN is 2-competitive in the new cost-accounting scheme, by letting Π be the optimal offline algorithm OFF.

COROLLARY 3. *On every sample path,*

$$\sum_{i=1}^{T} (o_i^{OLN} + d_i^{OLN} + \tilde{W}_i^{OLN}) \le 2 \sum_{i=1}^{T} (o_i^{OFF} + d_i^{OFF} + \tilde{W}_i^{OFF}).$$

Proof.

$$\begin{aligned} \sum_{i=1}^{T} (o_i^{\text{OLN}} + d_i^{\text{OLN}} + \tilde{W}_i^{\text{OLN}}) &\le 2 \max\left(\sum_{i=1}^{T} (o_i^{\text{OLN}} + d_i^{\text{OLN}}), \sum_{i=1}^{T} \tilde{W}_i^{\text{OLN}}\right) \\ &\le 2 \sum_{i=1}^{T} (o_i^{\text{OFF}} + d_i^{\text{OFF}} + \tilde{W}_i^{\text{OFF}}). \end{aligned}$$

□

Using Theorem 8, we can establish the performance guarantee of our cost-balancing algorithm in the original cost-accounting scheme.

COROLLARY 4.

$$E[\sum_{t=1}^{T}((s_t^{OLN}-m_t^{OLN})^\tau r+d_t^{OLN}+w^\tau f_t^{OLN})|\mathcal{F}]$$
$$\leq 2E[\sum_{t=1}^{T}((s_t^{OFF}-m_t^{OFF})^\tau r+d_t^{OFF}+w^\tau f_t^{OFF})|\mathcal{F}].$$

Proof.

$$\begin{aligned}
&E[\sum_{t=1}^{T}((s_t^{\mathrm{OLN}}-m_t^{\mathrm{OLN}})^\tau r+d_t^{\mathrm{OLN}}+w^\tau f_t^{\mathrm{OLN}})|\mathcal{F}]\\
&\quad=\sum_{i=1}^{n}s_{1i}q_i(r_i-1)+E[\sum_{t=1}^{T}((o_t^{\mathrm{OLN}}+d_t^{\mathrm{OLN}})+\tilde{w}_t^\tau f_t^{\mathrm{OLN}})|\mathcal{F}]\\
&\quad\leq 2\sum_{i=1}^{n}s_{1i}q_i(r_i-1)+2E[\sum_{t=1}^{T}((o_t^{\mathrm{OFF}}+d_t^{\mathrm{OFF}})+\tilde{w}_t^\tau f_t^{\mathrm{OFF}})|\mathcal{F}]\\
&\quad=2E[\sum_{t=1}^{T}((s_t^{\mathrm{OFF}}-m_t^{\mathrm{OFF}})^\tau r+d_t^{\mathrm{OFF}}+w^\tau f_t^{\mathrm{OFF}})|\mathcal{F}],
\end{aligned}$$

where the first and last equality follows from Theorem 8. □

Similar to the result in Section 3.4, we can show that in the allocation-scheduling model with cancellations, OLN has the best competitive ratio for any online algorithms.

THEOREM 10. *OLN is an optimal online algorithm for the allocation-scheduling model with cancellations.*

Proof. Theorem 5 implies that 2 is an upper bound of the optimal competitive ratio for the model with cancellations. Thus OLN is an optimal online algorithm for the model with cancellations. □

5. Numerical Performance

In this section we test the numerical performance of our online algorithm OLN. We vary multiple parameters to test the sensitivity of its performance. We compare the results against those of an optimal offline policy, an optimal stochastic policy which knows about the distribution of future

arrivals and capacities, and other heuristics. The results show that the gap between our online algorithm and the offline algorithm is within 16% in most cases when the total arrival rate is close to the daily capacity. The gap is larger when capacity exceeds demand or when demand exceeds capacity by a large amount. However, in such extreme cases certain naive policies perform very well and these policies should be used. For example, when there is a large surplus of capacity, no overtime resource should be used. When the number of arrivals is overwhelming, we always want to schedule as many jobs as possible in each period.

Our base test case is a two-class problem with parameters $w = (0.3, 0.1)$, $r = (2, 1.2)$ and $q = (0.1, 0.05)$. Arrivals are set to be stationary independent Poisson random variables with mean $\mathbf{E}[\delta_t] = (2, 3)$ for each period t. Capacity is constant $C_t = 5$ for all periods. The planning horizon has $T = 60$ periods and the discount factor is $\gamma = 0.95$. Each cost value is simulated by at least 10000 replicates. For each test case, we report the relative performance of the following five policies:

- V^{OLN} denotes the total cost of our online algorithm OLN.
- V^{OFF} denotes the optimal offline cost.
- V^{OPT} denotes the total cost of an optimal policy that knows the future demand distribution.
- V^{OLN^*} denotes the total cost of a variant OLN^* of our cost-balancing policy. OLN^* balances two cost components by using a different balancing ratio. An optimal balancing ratio is chosen for each test case.
- V_c denotes the total cost of a cutoff policy. In a cutoff policy, jobs are scheduled up to a certain threshold in each period. The threshold is optimized for each test case.

Table 2 shows the test results for different values of C_t. Our online scheduling policy OLN performs the best when demands and capacities are balanced, i.e., $C_t = \delta_1 + \delta_2 = 5$. It is interesting to notice that the gap between OPT and OFF is very small when C_t is below the total arrival rate, but increases to around 10% when C_t equals it. As mentioned earlier, this is because it is easy to carry out a near-optimal policy when C_t exceeds or is less than the total arrival rate by a large amount. Thus, it is most valuable and difficult to study the case when the daily capacity C_t is

close to the expected daily number of arrivals $\|\delta\|_1$. The result of our online policy is satisfactory in such situations. Its gap against OPT is only around 5%.

Tables 3 to 5 show the results when parameters of the higher priority class are varied. Generally all the scheduling policies we consider are not sensitive to these parameters. This is because most often all the higher priority jobs are served by regular capacities and thus do not affect the overtime and waiting costs. In Tables 6 and 7, the waiting costs of both classes are varied. In these cases, the performance of all the scheduling policies change broadly. Nevertheless, the variant OLN* of our online policy always outperforms the cutoff heuristic.

In Table 8, we allow the arrivals to be non-stationary and test different patterns of arrival rates. In the case of cyclic arrivals, the arrival rates are set to be $(4, 7)$, $(1, 1)$ and $(1, 1)$ for every three consecutive periods. The case of declining arrivals has rates dropping from $(3, 5)$ to $(1, 1)$ linearly, whereas the case of growing rates has the reversed pattern. In all these cases, the gap of our online policy is within 30%, and the gap of its variant OLN* is smaller than that of the cutoff policy by as much as 5%. Table 9 shows the results when there are more priority classes. In these settings, the waiting costs are uniformly distributed between 0.8 and 0.2, the cancellation probabilities are uniformly distributed between 0.2 and 0.05, and the cancellation costs are uniformly distributed between 8 and 1. The arrival rates for all the classes are set to be the same with $C_t = \|\delta\|_1$. We compare cost values against the cutoff heuristic. Both OLN and OLN* outperforms the cutoff policy. Moreover, the online policy performs better when there are more priority classes.

Table 2 Performance results under different values of C_t.

C_t	$V^{\mathrm{OPT}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}^*}/V^{\mathrm{OFF}}$	V_c/V^{OFF}
2	101.3%	137.1%	102.3%	102.3%
3	104.2%	135.2%	106.7%	107.6%
4	108.9%	128.0%	112.8%	120.3%
5	110.5%	115.5%	115.8%	118.1%
6	102.5%	115.3%	102.6%	102.6%
7	100.2%	116.5%	100.5%	100.2%

Table 3 Performance results under different values of w_1.

w_1	$V^{\mathrm{OPT}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}^*}/V^{\mathrm{OFF}}$	V_c/V^{OFF}
0.1	110.8%	115.4%	115.7%	117.6%
0.2	110.2%	115.5%	115.4%	117.4%
0.3	109.7%	115.6%	115.0%	117.3%
0.4	110.4%	115.6%	115.9%	118.5%
0.5	109.9%	115.6%	115.4%	118.5%
0.6	110.2%	115.6%	116.1%	119.2%
0.7	109.5%	115.7%	115.2%	118.9%
0.8	109.5%	115.7%	115.2%	119.5%
0.9	110.0%	115.6%	115.8%	120.5%

Table 4 Performance results under different values of q_1.

q_1	$V^{\mathrm{OPT}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}^*}/V^{\mathrm{OFF}}$	V_c/V^{OFF}
0.1	110.8%	115.5%	116.1%	118.5%
0.2	109.6%	115.5%	115.0%	117.5%
0.3	110.3%	115.6%	116.0%	118.7%
0.4	109.8%	115.6%	115.5%	118.6%
0.5	110.2%	115.8%	116.1%	119.4%
0.6	109.8%	115.7%	115.6%	119.4%
0.7	109.7%	115.7%	115.5%	119.7%
0.8	110.2%	115.7%	116.1%	120.7%
0.9	110.4%	115.7%	116.5%	121.3%

Table 5 Performance results under different values of r_1.

r_1	$V^{\mathrm{OPT}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}^*}/V^{\mathrm{OFF}}$	V_c/V^{OFF}
2	110.2%	115.6%	115.5%	117.8%
3	110.2%	115.5%	115.8%	118.2%
4	109.9%	115.7%	115.5%	118.4%
5	109.9%	115.6%	115.7%	118.8%
6	110.3%	115.6%	116.0%	119.7%
7	109.9%	115.7%	115.6%	119.8%
8	109.7%	115.6%	115.5%	120.0%

Table 6 Performance results when w_1 and w_2 are both increasing.

w_1	w_2	$V^{\mathrm{OPT}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}^*}/V^{\mathrm{OFF}}$	V_c/V^{OFF}
0.3	0.1	110.8%	115.5%	116.2%	118.5%
0.4	0.2	116.9%	127.4%	123.2%	133.9%
0.5	0.3	119.5%	132.9%	123.2%	129.3%
0.6	0.4	118.5%	136.7%	120.2%	121.2%
0.7	0.5	114.6%	138.7%	114.7%	114.7%
0.8	0.6	110.9%	141.6%	111.0%	111.0%
0.9	0.7	107.1%	143.0%	107.2%	107.2%

Table 7 Performance results when w_1 is increasing and w_2 is decreasing.

w_1	w_2	$V^{\mathrm{OPT}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}^*}/V^{\mathrm{OFF}}$	V_c/V^{OFF}
0.5	0.5	114.3%	138.8%	114.5%	114.7%
0.6	0.4	117.9%	136.7%	119.7%	120.6%
0.7	0.3	120.2%	132.9%	123.9%	130.0%
0.8	0.2	116.9%	127.4%	123.1%	134.3%
0.9	0.1	109.7%	115.7%	115.5%	120.2%

Table 8 Performance results when demand is non-stationary.

Demand Pattern	$V^{\mathrm{OPT}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}}/V^{\mathrm{OFF}}$	$V^{\mathrm{OLN}^*}/V^{\mathrm{OFF}}$	V_c/V^{OFF}
Cyclic	104.4%	111.7%	110.3%	111.0%
Declining	106.3%	130.2%	110.6%	115.9%
Growing	106.6%	121.5%	121.3%	121.6%

Table 9 Performance results under larger dimensions of state space.

Number of job classes	V^{OLN}/V_c	V^{OLN^*}/V_c
4	98.4%	91.6%
8	90.6%	88.0%

References

Albers, Susanne, Hiroshi Fujiwara. 2007. Energy-efficient algorithms for flow time minimization. *ACM Trans. Algorithms* **3**(4). doi:10.1145/1290672.1290686. URL http://doi.acm.org/10.1145/1290672.1290686.

Ayvaz, Nur, Woonghee Tim Huh. 2010. Allocation of hospital capacity to multiple types of patients. *J Revenue Pricing Manag* **9**(5) 386–398. URL http://dx.doi.org/10.1057/rpm.2010.30.

Ball, Michael O., Maurice Queyranne. 2009. Toward robust revenue management: Competitive analysis of online booking. *Operations Research* **57**(4) 950–963. doi:10.1287/opre.1080.0654. URL http://pubsonline.informs.org/doi/abs/10.1287/opre.1080.0654.

Bansal, Nikhil, DavidP. Bunde, Ho-Leung Chan, Kirk Pruhs. 2011. Average rate speed scaling. *Algorithmica* **60**(4) 877–889. doi:10.1007/s00453-009-9379-z. URL http://dx.doi.org/10.1007/s00453-009-9379-z.

Bansal, Nikhil, Ho-Leung Chan, Kirk Pruhs. 2009a. Speed scaling with an arbitrary power function. *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '09, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 693–701. URL http://dl.acm.org/citation.cfm?id=1496770.1496846.

Bansal, Nikhil, Ho-Leung Chan, Kirk Pruhs, Dmitriy Katz. 2009b. Improved bounds for speed scaling in devices obeying the cube-root rule. Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris Nikoletseas, Wolfgang Thomas, eds., *Automata, Languages and Programming*, *Lecture Notes in Computer Science*, vol. 5555. Springer Berlin Heidelberg, 144–155. doi:10.1007/978-3-642-02927-1_14. URL http://dx.doi.org/10.1007/978-3-642-02927-1_14.

Bansal, Nikhil, Tracy Kimbrel, Kirk Pruhs. 2007a. Speed scaling to manage energy and temperature. *J. ACM* **54**(1) 3:1–3:39. doi:10.1145/1206035.1206038. URL http://doi.acm.org/10.1145/1206035.1206038.

Bansal, Nikhil, Kirk Pruhs, Cliff Stein. 2007b. Speed scaling for weighted flow time. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '07, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 805–813. URL http://dl.acm.org/citation.cfm?id=1283383.1283469.

Blackburn, Joseph D. 1972. Optimal control of a single-server queue with balking and reneging. *Management Science* **19**(3) pp. 297–313. URL http://www.jstor.org/stable/2629512.

Borodin, Allan, Ran El-Yaniv. 1998. *Online Computation and Competitive Analysis*. Cambridge University Press.

Buchbinder, Niv, Tracy Kimbrel, Retsef Levi, Konstantin Makarychev, Maxim Sviridenko. 2013. Online make-to-order joint replenishment model: Primal-dual competitive algorithms. *Operations Research* **61**(4) 1014–1029. doi:10.1287/opre.2013.1188. URL http://pubsonline.informs.org/doi/abs/10.1287/opre.2013.1188.

Cardoen, Brecht, Erik Demeulemeester, Jeroen Beliën. 2010. Operating room planning and scheduling: A literature review. *European Journal of Operational Research* **201**(3) 921–932.

Carr, Scott, Izak Duenyas. 2000. Optimal admission control and sequencing in a make-to-stock/make-to-order production system. *Operations Research* **48**(5) 709–720. doi:10.1287/opre.48.5.709.12401. URL http://pubsonline.informs.org/doi/abs/10.1287/opre.48.5.709.12401.

Chan, Ho-Leung, Wun-Tat Chan, Tak-Wah Lam, Lap-Kei Lee, Kin-Sum Mak, Prudence W. H. Wong. 2007. Energy efficient online deadline scheduling. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '07, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 795–804. URL http://dl.acm.org/citation.cfm?id=1283383.1283468.

Chen, B., Chris N. Potts, Gerhard J. Woeginger. 1998. *A Review of Machine Scheduling: Complexity, Algorithms and Approximability*. Kluwer Academic Publishers.

Dellaert, N.P., M.T. Melo. 1998. Make-to-order policies for a stochastic lot-sizing problem using overtime. *International Journal of Production Economics* **5657**(0) 79 – 97. doi:http://dx.doi.org/10.1016/S0925-5273(98)00053-X. URL http://www.sciencedirect.com/science/article/pii/S092552739800053X. Production Economics: The Link Between Technology And Management.

Denton, Brian T., Andrew J. Miller, Hari J. Balasubramanian, Todd R. Huschka. 2010. Optimal allocation of surgery blocks to operating rooms under uncertainty. *Operations Research* **58**(4-part-1) 802–816. doi:10.1287/opre.1090.0791. URL http://pubsonline.informs.org/doi/abs/10.1287/opre.1090.0791.

Elmachtoub, A. N., R. Levi. 2014. Supply chain management with online customer selection. Working paper.

Federgruen, Awi, Kut C. So. 1990. Optimal maintenance policies for single-server queueing systems subject to breakdowns. *Operations Research* **38**(2) pp. 330–343. URL http://www.jstor.org/stable/171342.

Gerchak, Yigal, Diwakar Gupta, Mordechai Henig. 1996. Reservation planning for elective surgery under uncertain demand for emergency surgery. *Management Science* **42**(3) pp. 321–334. URL http://www.jstor.org/stable/2634346.

Gocgun, Yasin, Archis Ghate. 2012. Lagrangian relaxation and constraint generation for allocation and advanced scheduling. *Computers & Operations Research* **39**(10) 2323 – 2336. doi:http://dx.doi.org/10.1016/j.cor.2011.11.017. URL http://www.sciencedirect.com/science/article/pii/S030505481100342X.

Greenhouse, Steven. 2012. A part-time life, as hours shrink and shift. *The New York Times* .

Guerriero, Francesca, Rosita Guido. 2011. Operational research in the management of the operating theatre: a survey. *Health care management science* **14**(1) 89–114.

Gupta, D. 2007. Surgical suites' operations management. *Production and Operations Management* **16**(6) 689–700.

Huh, Woonghee Tim, Nan Liu, Van-Anh Truong. 2013. Multiresource allocation scheduling in dynamic environments. *Manufacturing & Service Operations Management* **15**(2) 280–291. doi:10.1287/msom.1120.0415. URL http://pubsonline.informs.org/doi/abs/10.1287/msom.1120.0415.

Karlin, Anna R., Mark S. Manasse, Lyle A. McGeoch, Susan Owicki. 1990. Competitive randomized algorithms for non-uniform problems. *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '90, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 301–309. URL http://dl.acm.org/citation.cfm?id=320176.320216.

Karlin, AnnaR., MarkS. Manasse, Larry Rudolph, DanielD. Sleator. 1988. Competitive snoopy caching. *Algorithmica* **3**(1-4) 79–119. doi:10.1007/BF01762111. URL http://dx.doi.org/10.1007/BF01762111.

Keskinocak, Pinar, R. Ravi, Sridhar Tayur. 2001. Scheduling and reliable lead-time quotation for orders with availability intervals and lead-time sensitive revenues. *Management Science* **47**(2) 264–279. doi:10.1287/mnsc.47.2.264.9836. URL http://dx.doi.org/10.1287/mnsc.47.2.264.9836.

Levi, R., R. O. Roundy, D. B. Shmoys, V. A. Truong. 2008a. Approximation algorithms for capacitated stochastic inventory control models. *Operations Research* **56**(5) 1184–1199.

Levi, Retsef, Ganesh Janakiraman, Mahesh Nagarajan. 2008b. A 2-approximation algorithm for stochastic inventory control models with lost sales. *Mathematics of Operations Research* **33**(2) 351–374.

Levi, Retsef, Martin Pál, Robin Roundy, David B. Shmoys. 2005. Approximation algorithms for stochastic inventory control models. Michael J?nger, Volker Kaibel, eds., *Integer Programming and Combinatorial Optimization*, *Lecture Notes in Computer Science*, vol. 3509. Springer Berlin / Heidelberg, 306–320.

Lin, Minghong, Zhenhua Liu, A Wierman, L.L.H. Andrew. 2012. Online algorithms for geographical load balancing. *Green Computing Conference (IGCC), 2012 International*. 1–10. doi:10.1109/IGCC.2012.6322266.

Martin, G. E., Joyce L. Grahn, Lyn D. Pankoff, Laurence A. Madeo. 1992. A mechanism for reducing small-business customer waiting-line dissatisfaction. *Managerial and Decision Economics* **13**(4) 353–361. doi:10.1002/mde.4090130410. URL `http://dx.doi.org/10.1002/mde.4090130410`.

May, Jerrold H, William E Spangler, David P Strum, Luis G Vargas. 2011. The surgical scheduling problem: Current research and future opportunities. *Production and Operations Management* **20**(3) 392–405.

Min, Daiki, Yuehwern Yih. 2010. An elective surgery scheduling problem considering patient priority. *Computers & Operations Research* **37**(6) 1091 – 1099. doi:http://dx.doi.org/10.1016/j.cor.2009.09.016. URL `http://www.sciencedirect.com/science/article/pii/S0305054809002342`.

Noga, John, Steven S. Seiden. 2001. An optimal online algorithm for scheduling two machines with release times. *Theoretical Computer Science* **268**(1) 133 – 143. doi:http://dx.doi.org/10.1016/S0304-3975(00)00264-4. URL `http://www.sciencedirect.com/science/article/pii/S0304397500002644`. Online Algorithms '98.

Özdamar, Linet, Tülin Yazgaç. 1997. Capacity driven due date settings in make-to-order production systems. *International Journal of Production Economics* **49**(1) 29 – 44. doi:http://dx.doi.org/10.1016/S0925-5273(96)00116-8. URL `http://www.sciencedirect.com/science/article/pii/S0925527396001168`.

Patrick, Jonathan, Martin L. Puterman, Maurice Queyranne. 2008. Dynamic multipriority patient scheduling for a diagnostic resource. *Operations Research* **56**(6) 1507–1525. doi:10.1287/opre.1080.0590. URL `http://pubsonline.informs.org/doi/abs/10.1287/opre.1080.0590`.

Rubino, Melanie, Bar Ata. 2009. Dynamic control of a make-to-order, parallel-server system with cancellations. *Operations Research* **57**(1) 94–108. doi:10.1287/opre.1080.0532. URL `http://dx.doi.org/10.1287/opre.1080.0532`.

Stidham, Jr., S. 1985. Optimal control of admission to a queueing system. *Automatic Control, IEEE Transactions on* **30**(8) 705–713. doi:10.1109/TAC.1985.1104054.

Truong, Van-Anh. 2014a. Approximation algorithm for the stochastic multiperiod inventory problem via a look-ahead optimization approach. *Mathematics of Operations Research* doi:10.1287/moor.2013.0639. URL `http://dx.doi.org/10.1287/moor.2013.0639`.

Truong, Van-Anh. 2014b. Optimal advance scheduling. Working paper.

Wagner, Michael R. 2010. Fully distribution-free profit maximization: The inventory management case. *Mathematics of Operations Research* **35**(4) 728–741. doi:10.1287/moor.1100.0468. URL `http://pubsonline.informs.org/doi/abs/10.1287/moor.1100.0468`.

Xiong, Wei, David Jagerman, Tayfur Altiok. 2008. queue with deterministic reneging times. *Performance Evaluation* **65**(34) 308 – 316. doi:http://dx.doi.org/10.1016/j.peva.2007.07.003. URL `http://www.sciencedirect.com/science/article/pii/S016653160700079X`.

Yao, F., A. Demers, S. Shenker. 1995. A scheduling model for reduced cpu energy. *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*. 374–382. doi:10.1109/SFCS.1995.492493.

Zhang, Liqi, Lingfa Lu, Jinjiang Yuan. 2009. Single machine scheduling with release dates and rejection. *European Journal of Operational Research* **198**(3) 975 – 978. doi:http://dx.doi.org/10.1016/j.ejor.2008.10.006. URL `http://www.sciencedirect.com/science/article/pii/S0377221708008345`.