Robust Appointment Scheduling with Heterogeneous Costs

Andreas S. Schulz
Technische Universität München, Germany
andreas.s.schulz@tum.de

Rajan Udwani
Columbia University, New York, NY, USA
rudwani@alum.mit.edu

Abstract

Designing simple appointment systems that under uncertainty in service times, try to achieve both high utilization of expensive medical equipment and personnel as well as short waiting time for patients, has long been an interesting and challenging problem in health care. We consider a robust version of the appointment scheduling problem, introduced by Mittal et al. (2014), with the goal of finding simple and easy-to-use algorithms. Previous work focused on the special case where per-unit costs due to under-utilization of equipment/personnel are homogeneous i.e., costs are linear and identical. We consider the heterogeneous case and devise an LP that has a simple closed-form solution. This solution yields the first constant-factor approximation for the problem. We also find special cases beyond homogeneous costs where the LP leads to closed form optimal schedules. Our approach and results extend more generally to convex piece-wise linear costs.

For the case where the order of patients is changeable, we focus on linear costs and show that the problem is strongly NP-hard when the under-utilization costs are heterogeneous. For changeable order with homogeneous under-utilization costs, it was previously shown that an EPTAS exists. We instead find an extremely simple, ratio-based ordering that is 1.0604 approximate.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Discrete optimization; Theory of computation → Scheduling algorithms

Keywords and phrases Appointment scheduling, approximation algorithms, robust optimization

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2019.25

Category APPROX

Funding *Andreas S. Schulz*: Alexander von Humboldt Foundation (with funds from BMBF)
Rajan Udwani: ONR Grant N00014-17-1-2194, NSF Grant CMMI 1636046

Acknowledgements The authors would like to thank James B. Orlin for helpful discussions.

1 Introduction

Consider the problem of scheduling appointments in service operations where customers are served sequentially by a single server. Service times of customers are uncertain, and we wish to assign time slots for serving the customers in advance. An important practical setting where this problem arises everyday is in health care services, where there are numerous instances that require efficient scheduling of appointments, such as scheduling outpatient appointments in primary care and specialty clinics, scheduling surgeries for operating rooms, or appointments for MRI scans. Often in these settings the order in which patients undergo the procedure is known in advance. Then a day before the procedures, a hospital manager determines planned start times and how much time to allot to each procedure. If the manager allots too small an interval to a procedure, it could easily go overtime and delay the next procedure. The inconvenience and costs resulting from such a delay are referred to as the *overage* cost for that procedure. On the contrary, if the manager assigns a very large interval, then the procedure will likely end much sooner. In this case the hospital incurs an *underage* cost as the equipment and personnel may be left idle until the scheduled start of the next procedure. We would like to design an appointment schedule that can achieve the desired trade-off between overage and underage costs.

In an influential survey on outpatient scheduling, Cayirli and Veral [6] concluded that the "biggest challenge for future research will be to develop easy-to-use heuristics". Traditionally, most models for the problem are stochastic in nature and assume distributional knowledge of the uncertain service times. While such models are very powerful, estimating distributions accurately can require large amounts of data. There are many settings where such data is in fact available but, in applications related to health care appointment scheduling there is evidence (ref. [17, 18, 8]) that the amount of data available by surgery types, let alone by surgery types and surgeons, is extremely limited. Moreover, computing an objective function that involves finding the expectation of a high dimensional non-linear function can be computationally burdensome. Further, Mak et al. [18] and Mittal et al. [20] point out that methods proposed for solving stochastic models often use sophisticated subroutines, such as submodular function minimization and Monte-Carlo techniques, that may not always be practical. Robust optimization (e.g., [4, 5, 3]) offers an alternative framework to address uncertainty that avoids distributional assumptions. Instead, it uses deterministic uncertainty sets and performs a worst-case analysis w.r.t. to the uncertainty. This addresses the problems arising out of insufficient data and often these models are more tractable.[1] Indeed, for the case of appointment scheduling, Mittal et al. [20] introduced the following robust formulation for the problem.

Referring to the procedures/services in any given context simply as "jobs", in the robust appointment scheduling problem we are given n jobs with uncertain service times $\{t_i\}$. Assume that the time t_i for job i can be anything in the range $[p_i - \hat{\delta}_i, p_i + \delta_i]$. The range is known to us but the value of t_i can be chosen adversarially. Our task is to propose an appointment start time for each job. A job will be available to process after this start time and jobs will be served in order of increasing index i. The order in which jobs are processed is fixed a priori. Due to uncertainty in service times, a job may have to wait to be processed after its appointed start time. The overage cost incurred per unit of wait is given by $o_i \geq 0$. Similarly, $u_i \geq 0$ is the per-unit underage cost associated with job i. Given a set of appointment start times, an adversary chooses the worst possible instantiation of service durations t_i i.e., one that maximizes the overall overage/underage cost. We seek start times to minimize this worst case total cost.

Mittal et al. [20] found closed-form optimal solutions for the special case of homogeneous underage costs i.e., $u_i = u$ for all jobs i. We study more general cases[2], where a direct generalization of their solution can be arbitrarily bad. Our central goal here is to find simple, easy to implement, and theoretically well founded algorithms. In the following we summarize our contributions and discuss closely related work.

---

[1] Sometimes at the expense of being too conservative.

[2] Arguably, just as processing times and uncertainty vary across procedures, so can personnel and equipment. Thus, in general the per-unit underage costs will vary across procedures. Moreover, it is also reasonable to assume that the per-unit cost changes with the amount of underage/overage. For instance, keeping patients waiting for another unit of time becomes less and less desirable with every unit of delay. This prompts the study of piece-wise linear overage and underage costs.

Robust Appointment Scheduling (RAS). We explore the structure of optimal appointment schedules and find several properties that all optimal schedules must satisfy. Somewhat surprisingly, we find that in every optimal appointment schedule, the case of all jobs underaged as well as the case of all jobs overaged are always worst-cases for the adversary's problem. Casting these in the form of linear (in)equalities gives an LP, resulting in a 2-approximation. Further simplification yields a closed-form solution to the LP that could be implemented even on a spreadsheet. More generally, when the per-unit costs are allowed to change with the amount of overage/underage and the cost functions are piece-wise linear and convex (and non-decreasing), we are able to generalize these properties to obtain a 2-approximation for the problem via solving a min-cost flow problem on a graph with convex piece-wise linear arc costs.

Previously, Mittal et al. [20] examined the special case of identical underage costs and, quite remarkably, found a closed-form optimal solution in that case. While their analysis was tailored for the special case, our different approach allows us to characterize exactly when such a result holds, and we discover more general conditions under which there exists a closed-form optimal solution to the problem. We also show a similar extension for the case of convex piece-wise linear costs, where we give a simple iterative algorithm that gives an optimal allocation under additional assumptions on the cost functions.

Robust Appointment Scheduling and Ordering (RASO). While the order of patients is often fixed in advance, there are instances where patient order is flexible and simultaneously part of the decision making [6]. When underage costs are identical, Mittal et al. [20] established a key connection between RASO and the theory of scheduling jobs on a single machine to minimize the sum of a weighted nonlinear (concave) cost function of the completion times [19, 12, 25]. Focusing on the special case, we exploit this connection further to give an extremely practical ratio based ordering policy, inspired by Smith's rule [24]. We call it the Customized-Smith rule (short C-Smith) and show that it is 1.0604-approximate. Further, we find an algorithm that is as good as C-Smith on every instance, but is also locally optimal w.r.t. pairwise swaps of neighboring jobs. Previously, Mittal et al. [20] showed that using the EPTAS for the related min-sum scheduling problem given by Stiller and Wiese [25], yields an EPTAS for RASO. The runtime of the EPTAS scales as $O(2^{1/\epsilon^7}n)$, and its implementation complexity makes it hard to use in practice. Mittal et al. [20] offered Smith's rule as a practical alternative, with approximation ratio 1.137 due to a result in Höhn and Jacobs [12]. Finally, for the general case (arbitrary underage costs), we show the problem is strongly NP hard. We also briefly discuss the inefficacy of *list ordering* heuristics for the problem and give a heuristic with a matching lower bound.

Other related work. The existing literature on appointment scheduling is quite diverse and, in addition to the robust model we discussed above, includes numerous results on stochastic optimization models, queueing models, as well as distributionally robust optimization models. We only give a very brief review here and refer the reader to [6] for a broad overview of the challenges in scheduling appointment systems in health care, and to [18, 20] for more comprehensive surveys on past work. Starting with more recent work, Jiang et al. [14] consider a distributionally robust model and propose a MINLP formulation that can handle random no-shows. Mak et al. [18] consider a distributionally robust model given marginal moments for job durations, and formulate the problem as a tractable conic program. For the case of flexible job order, they show that under certain assumptions a simple variance based ordering can be optimal. Prior to [18], Kong et al. [16] first considered a distributionally robust formulation but with cross-moments as opposed to marginal moments. They formulate the problem as a copostive cone program and propose a tractable SDP relaxation. Wang [26] modeled the problem using a queueing model where the processing times of the jobs are i.i.d. exponential and new jobs may be released over time. Wang [27] generalized the model to allow for different mean processing times for jobs. In a different direction, the problem was modeled as a two-stage stochastic linear program in Denton and Gupta [8]. For this problem, Robinson and Chen [23] compute near-optimal solutions using a Monte-Carlo integration technique. Kaandorp and Koole [15] considered a local search algorithm and proved that it converges to an optimal solution. Another stochastic model was introduced by Green et al. [11]. They considered the problem of outpatient appointment scheduling with emergency services and modeled it as a dynamic stochastic control problem. In highly influential work, Begen and Queyranne [2] considered a discrete stochastic model (where job durations are integer random variables with finite support). They showed that the problem reduces to an instance of submodular function minimization, under certain assumptions on the per-unit costs. Begen et al. [1] extended the idea, proving a similar result for a data-driven discrete stochastic model. Ge et al. [10] further extended the result of Begen and Queyranne [2], to the setting of convex piece-wise linear per-unit costs.

Assumptions on underage costs. Previous work on the stochastic and distributionally robust models discussed above, often assumes an upper bound on the variation in underage costs. Formally, consider jobs that are indexed in the order they are scheduled. So we have jobs $i \in \{1, \dots, n\}$,with per unit costs u_i, o_i. Now for instance, the result by Begen and Queyranne [2], assumes that there exists values $\alpha_i \leq o_i$ such that $u_{i+1} \leq u_i + \alpha_i - \alpha_{i+1}$ for every $i \in [n-1]$. Descending underage costs for example, satisfy this condition. More generally, [14, 18, 16, 10] assume that $u_{i+1} \leq o_i + u_i$ for every $i \in [n-1]$ (or that $u_i = u$ for all i). In effect, these assumptions are satisfied as long as the underage costs don't *increase* too sharply.

In contrast, we give a closed form optimal allocation for instances where the u_i values do not *decrease* too sharply. For example, our results yield a closed form optimal solution for the case of non-decreasing underage costs. This includes the case $u_{i+1} \geq u_i + o_i \, \forall i \in [n-1]$, converse of the condition assumed in some previous work. If the per-unit underage costs are decreasing, we show that as long as the per-unit values are "large enough", our LP still leads to a closed form optimal allocation. In practical terms, it is quite possible that the underage costs could be increasing or decreasing, preventing a direct comparison of our assumptions with past work. In fact, for the RASO problem where one has the flexibility to choose the order of jobs, an optimal order can have arbitrarily varying underage costs. From a technical viewpoint, our results complement previous work, indicating that perhaps the robust model is more tractable in instances where other models are not, and vice versa.

Overview of the paper. In Section 2, we introduce notation and formally state the problem. We also make certain observations that simplify the problem w.l.o.g.. Then in Section 3, we explore the appointment scheduling problem under fixed order. We first give our LP based approximation for general underage costs in Section 3.1. And later in Section 3.2, we tighten our LP formulation to obtain optimal closed form solutions under additional assumptions on the costs. In Section 4, we consider the problem of jointly finding an optimal order as well as appointment schedule (RASO). We show that the problem is NP hard in general and discuss the limitations of a classes of simple heuristics in Section 4.1. In Section 4.2, we focus on the ordering and scheduling problem in the special case of homogeneous underage costs. Here we discuss some disadvantages of existing results and propose two new heuristics to tackle some of the issues. Finally, we conclude with some open problems in Section 5.

2 Notation & Preliminaries

We start with a description of RAS, and define additional notation required for RASO in Section 4. Recall, we have n jobs to be served in order of increasing index $i \in \{1, \dots, n\}$. Service times are uncertain and modeled via a box uncertainty set: job i takes time t_i in the range $[p_i - \hat{\delta}_i, p_i + \delta_i]$. Here we assume that $0 \leq \hat{\delta}_i \leq p_i$ as well as $\delta_i \geq 0$, for all jobs i. We denote by $o_i \geq 0$ and $u_i \geq 0$ the per-unit overage cost and per-unit underage cost of job i, respectively. More generally, we represent overage and underage costs as *non-decreasing* functions $o_i(\cdot), u_i(\cdot)$ respectively. The case of constant per-unit costs is then given by $o_i(x) = o_i \cdot x$ and $u_i(x) = u_i \cdot x$. We would like to appoint job start times, $\{A_i\}_i$, such that every job arrives at its appointed time and is served as soon as possible after. Given $\{A_i\}_i$, consider an arbitrary instance of service times $\{t_i\}_i$, and let $\{C_i\}_i$ denote the completion times of jobs.[3] If job i is delayed and ends after the appointed start time for job $i+1$, we incur overage cost $o_i(C_i - A_{i+1})$. Similarly, if job i ends before the appointed start time for job $i+1$, we incur underage cost $u_i(A_{i+1} - C_i)$. Therefore, the cost due to job i is $\max\{o_i(C_i - A_{i+1}), u_i(A_{i+1} - C_i)\}$. The RAS problem can now be stated as follows,

$$\min_{\{A_i\}_i} \underbrace{\left(\max_{t_i \in [p_i - \hat{\delta}_i, p_i + \delta_i] \, \forall i \in [n]} \sum_{i=1}^{n} \max\{o_i(C_i - A_{i+1}), u_i(A_{i+1} - C_i)\} \right)}_{\textit{Adversary's problem, given appointment times } A_i.} . \tag{1}$$

The adversary's problem in (1), finds a *worst possible profile/instance of service times* $T = (t_i) = (t_1, \dots, t_n)$, maximizing the cost, given the schedule $\{A_i\}_i$. Let $c(T, A, i) = \max\{o_i(C_i(T) - A_{i+1}), u_i(A_{i+1} - C_i(T))\}$ denote the cost of job i given allocation $\{A_i\}_i$ and time profile T. When the allocation is clear from the context, we use the shorthand $c(T, i)$. Let $c(T) = \sum_i c(T, i)$ denote the total cost due to profile T. It is not difficult to see that an optimal allocation needs to allocate at least $p_i - \hat{\delta}_i$ time for job i, so we let $p_i - \hat{\delta}_i = 0$ for every job i, w.l.o.g. (also in Lemma 6 of [20]). To simplify notation henceforth, we let service times be in the range $[0, \Delta_i]$, where $\Delta_i = \hat{\delta}_i + \delta_i$. Now, given appointment schedule $\{A_i\}_i$, consider equivalent variables x_i that represent the duration allocated for a job. We have, $x_i = A_{i+1} - A_i$, which is the difference between the start times of job $i+1$ and job i. Equivalently, $A_i = \sum_{j=1}^{i-1} x_j$ for $i \geq 2$ and $A_1 = 0$, w.l.o.g. We call $\{x_i\}_i$, and sometimes by abuse of terminology $\{A_i\}_i$, the *allocation*. Note that in our model, job n is cost free since there is no appointment succeeding it ($A_{n+1} = \infty$). W.l.o.g. we may assume that job n also suffers from overage and underage based on its assigned end time (assign a dummy job to succeed job n). Note, if $u_i(\cdot) = 0$, we allot a very large time slot for job i and jobs on different sides of the slot become independent. Therefore, we let $u_i(\cdot) \neq 0$ for all i, w.l.o.g.. Also, if $o_n(\cdot) = 0$ we can assume that $x_n = 0$ and in fact ignore job n, therefore we also let $o_n(\cdot) \neq 0$, w.l.o.g.. To coalesce, the assumptions we have made w.l.o.g., so far:

(i) Service time for job i lies in $[0, \Delta_i]$; (ii) for every i, $u_i(\cdot) \neq 0$ ($u_i > 0$ in the constant per-unit case); and (iii) $o_n(\cdot) \neq 0$ ($o_n > 0$ in the constant per-unit case).

Finally, we say job i is *underaged* if it ends on or before time A_{i+1}, and *overaged* otherwise. However, if $\Delta_i = 0$ and job i starts/ends at A_{i+1}, we consider it to be both overaged and underaged (for technical reasons). Further, if job i takes time Δ_i to be served, we say it runs for *maximum time*. Note that if a job i is underaged in some worst-case $T = (t_i)$, then w.l.o.g., $t_i = 0$, as the underaged costs $u_i(\cdot)$ are non-decreasing functions for every i (similar

[3] $C_i = \max\{C_{i-1}, A_i\} + t_i$ for all $i \geq 1$ with $C_0 = A_1 = 0$, w.l.o.g.

to Lemma 11 of [20]). Observe also that if a job i is underaged in some worst-case time profile, and it has zero cost, then i must end at A_{i+1} and take zero time. Otherwise, we have a strictly worse case by underaging i (simply reduce the time taken by i by some small $\epsilon > 0$).

Finally, let $S_i = \{i, i+1, \ldots, n\} = \{j | j \geq i\}$ be the subset of the last $n - i + 1$ jobs in the schedule. Let x_i^S denote the optimal time allocation for job i when considering only jobs in a subset S that contains i. When $S = [n]$ we often use shorthand x_i. The next section discusses a result from previous work and offers some intuition for our treatment of the general case that follows subsequently in Section 3.

2.1 Closed Form Optimal Solution of Mittal et al. [20]

Consider a single job, with per unit costs u, o and maximum time Δ. If this job is alloted time duration x, then the worst case cost is given by $\max\{u(x), o(\Delta - x)\}$, minimized at $x = \Delta \frac{o}{o+u}$ (recall, we assume that a dummy job always follows the last job, so the last job incurs overage cost for delays). Mittal et al. [20] showed that for constant per-unit underage costs, $u_i = u$ for every job $i \in [n]$, this formula generalizes and the optimal allocation is given simply by, $x_i = \Delta_i \frac{o(i)}{o(i)+u}$. Here $o(i)$ is the sum $\sum_{j=i}^{n} o_j$, of the per unit overage costs of the jobs succeeding job i. So each job is alloted a fraction of its maximum service time Δ_i, with a smaller fraction for larger values of u (to prevent large underage costs). Further, earlier jobs are alloted a larger fraction of their maximum time to prevent a large cascade of delays for jobs serviced later on.

Now, consider a natural generalization of this formula for heterogeneous underage costs given by, $x_i = \Delta_i \frac{o(i)}{o(i)+u_i}$. Unfortunately, this can be a suboptimal allocation even for two jobs with reasonable parameter values, and in general an arbitrarily bad approximation. For example, consider two jobs that are almost identical except that job 2 (which is scheduled later) has a small underage cost. Specifically, let $\Delta_1 = \Delta_2 = 1$ and suppose per unit costs $o_1 = o_2 = 1$ and $u_1 = M \geq 2$, but $u_2 = 1$. The allocation given by the formula sets x_1 to $\frac{2}{M+2}$ and x_2 to $\frac{1}{2}$. The worst case cost of this allocation is attained when both jobs are underaged or both overaged (more on this later), and equals $\frac{2M}{M+2} + \frac{1}{2}$. For large M, this value approaches $\frac{5}{2}$. Now instead, consider the following allocation, $y_1 = 0$ and $y_2 = \Delta_2 \frac{o_2}{o_2+u_2} + \Delta_1 \frac{o(1)}{o_2+u_2} = \frac{1}{2} + 1$ (we show later in Section 3.2, that this is in fact an optimal allocation). It is easily checked that the worst case cost is $\frac{3}{2}$ for this allocation. More generally, if we set o_1 and u_2 to a small value ϵ, the first allocation becomes arbitrarily bad. Intuitively, this demonstrates that if jobs that are later in the order have small underage costs, it is beneficial to allocate larger time to these later jobs to buffer for delays from earlier jobs. When this is not the case (such as for homogeneous underage costs), it is better to instead buffer by allocating larger fractions to jobs that are earlier in the order.

3 Robust Appointment Scheduling

3.1 Heterogeneous Per-unit Underage Costs

In this section, we focus on the case of constant per-unit costs o_i, u_i. The key behind our results lies in finding useful properties satisfied by optimal allocations. Towards that end, it will be instrumental to understand the worst cases (solutions to the adversaries' problem) for optimal allocations. Recall the adversary's objective: given $\{A_i\}_i$, maximize cost $c(T, A)$ over all possible time profiles T allowed by the uncertainty set. It turns out that for an arbitrary allocation, there could be a unique worst-case where some jobs are overaged while others are underaged, and for some jobs the service time t_i is neither 0 nor Δ_i [4]. It turns out though, that for any given allocation the adversary's problem can be reduced to an instance of finding longest paths on a directed acyclic graph with $n+1$ nodes [21], and therefore can be solved in polynomial time. However, using this fact to find an optimal allocation does not obviously lead to a tractable problem. Instead, here we shall find properties in the form of linear inequalities that every optimal allocation satisfies. Using these, we formulate an LP relaxation for the problem and show that an optimal solution to the LP is 2-approximate.

As a natural next step, we then look for more structure to further tighten the formulation. Surprisingly, we find that the adversary's problem given an *optimal allocation* for problem (1) is easily solved; in every optimal allocation, $t_i = \Delta_i$ for every i (all jobs taking maximum time) and $t_i = 0$ for all i (all jobs taking zero time) are worst-cases. Combining this with other structural insights, in Section 3.2 we propose a strengthened formulation that leads to closed form optimal solutions under some assumptions on underage costs. In the lemmas that follow we introduce two properties of optimal allocations, leading to the first LP formulation.

▶ **Lemma 1.** *For every optimal allocation $\{x_i\}_i$, $\sum_{j=1}^k x_j \leq \sum_{j=1}^k \Delta_j$ for all $k = 1, \dots, n$.*

Proof. Given an optimal allocation $\{x_i\}_i$, let $\delta_k = \sum_{j=1}^k x_j - \sum_{j=1}^k \Delta_j$, for $k \in [n]$. Suppose, $\delta_k > 0$ for some k, and let k_0 be the smallest such k. Given this, notice that even if the first k_0 jobs take maximum time, job k_0 can never be overaged. Therefore, decreasing x_{k_0} decreases the underage cost of job k_0 ($u_{k_0} > 0$) and thus, also the worst-case cost. So the allocation $\{x'_i\}_i$ where, $x'_{k_0} = x_{k_0} - \delta_{k_0}$ and $x'_i = x_i$ for all other i is clearly a better allocation, contradiction. This lemma also follows as a direct corollary of Lemma 5 stated later on. ◀

▶ **Lemma 2.** *Given an allocation $\{x_i\}_i$ where $\sum_{j=1}^i x_j \leq \sum_{j=1}^i \Delta_j$ for all i, and a time profile $T = (t_i)$. The cost $c(T)$ of profile T is at most $\sum_{i=1}^n o(i)(\Delta_i - x_i) + \sum_{i=1}^n u_i x_i$.*

Proof. For any given time profile T, job i is either underaged with $c(T, i) \leq u_i x_i$ or overaged with cost at most $o_i \sum_{j \leq i}(\Delta_j - x_j)$. The latter follows from the fact that $C_i \leq \sum_{j \leq i} \Delta_j$ since $\sum_{j=1}^k x_j \leq \sum_{j=1}^k \Delta_j$ for all $k \in [i]$. Therefore, $\sum_{i=1}^n c(T, i) \leq \sum_{i=1}^n o_i \sum_{j=1}^i (\Delta_j - x_j) + \sum_{i=1}^n u_i x_i$. Rearranging the sum we have, $\sum_{i=1}^n o_i \sum_{j=1}^i (\Delta_j - x_j) = \sum_{j=1}^n (\Delta_j - x_j) o(j)$. ◀

Using the above results, the following LP now gives us an extremely simple approximation for the general problem that could be easily implemented on most systems.

$$\begin{aligned} \textbf{LP-1:} \quad & \min \quad \sum_{j=1}^n (u_j - o(j)) y_j \\ & \text{s.t.} \quad \sum_{j=1}^k (y_j - \Delta_j) \leq 0 \quad \forall k \in [n]; \\ & \qquad y_j \geq 0 \quad \forall j \in [n]. \end{aligned} \tag{2}$$

▶ **Theorem 3.** *LP-1 is a (tight) 2-approximation, and the following is an optimal solution to LP-1. Define $m_i = \arg\min_{j \geq i}(u_j - o(j))$, then for every i,*

$$y_i = \begin{cases} 0 & \text{if } u_i - o(i) \geq 0 \text{ or } i \neq m_i \\ \sum_{j | m_j = i} \Delta_j & \text{otherwise.} \end{cases}$$

[4] This is in contrast to Lemma 9 in [20] for the special case of $u_i = u \, \forall i$.

Proof. The proof of optimality for the proposed solution is easy to verify. To see the guarantee let us re-write the LP-1 objective as,

$$\min \sum_j u_j y_j + \sum_j o(j)(\Delta_j - y_j).$$

From constraints (2) and Lemma 2, this is an upper bound on the worst-case cost of any feasible solution to LP-1. Let $\{y_i^*\}$ denote an optimal solution to the LP. Lemma 1 implies that every optimal allocation $\{x_i\}$ is a feasible solution to LP-1. Further, $\sum_j u_j x_j$ denotes the cost when all jobs take zero time and $\sum_j o(j)(\Delta_j - x_j)$ is the cost of overaging all jobs with each job taking maximum time (Δ_i). Therefore, the worst-case cost of an optimal allocation, denoted OPT, is at least $\max\{\sum_j u_j x_j, \sum_j o(j)(\Delta_j - x_j)\}$ (we show later that these two costs are both in fact, equal to OPT). Therefore,

$$\sum_j u_j y_j^* + \sum_j o(j)(\Delta_j - y_j^*) \leq \sum_j u_j x_j + \sum_j o(j)(\Delta_j - x_j) \leq 2\ OPT.$$

For a tight instance, consider two jobs $\{1, 2\}$ with $\Delta_1 = \epsilon \to 0$, $u_1 = 1/\epsilon, o_1 = 1/\epsilon - (1+\epsilon)$ and $\Delta_2 = u_2 = o_2 = 1$. Therefore, $\frac{u_2}{u_2+o_2} = 0.5 \lessapprox \frac{u_1}{u_1+o(1)}$. Consider the allocation $x_2 = 0.5$ and $x_1 = \epsilon \frac{o(1)}{u_1+o(1)} \approx \epsilon/2$. It is easy to see that the worst-case of the allocation is when both jobs are underaged (overaged) and hence the cost of this allocation is $u_1 x_1 + u_2 x_2 \approx 1$. Now, consider the solution $y_2 = 0.5 + \frac{\Delta_1 o(1)}{u_2+o_2} \approx 1$ and $y_1 = 0$. This is an optimal solution to the LP and the worst-case occurs when job 2 is underaged and job 1 is overaged. The worst-case cost of this allocation is $o_1\Delta_1 + u_2(y_2 - \Delta_1) \approx 2$. ◀

Beyond Constant Per-unit Costs

Suppose instead of scalar costs o_i, u_i, we have non-decreasing, piece-wise linear and convex cost functions $o_i(\cdot), u_i(\cdot)$. Then it is easy to check that Lemma 1 still holds. Similar to Lemma 2, we have that given an allocation $\{x_i\}_i$, satisfying Lemma 1, the cost $c(T)$ for any profile T is at most,

$$\sum_i u_i(x_i) + \sum_i o_i\Big(\sum_{j\leq i}(\Delta_j - x_j)\Big).$$

Now the problem of minimizing this objective subject to the linear constraints in LP-1, is a min cost flow problem with arc costs given by the overage and underage cost functions. More specifically, consider a directed graph with $n+1$ nodes and $2n$ edges, where there is a directed edge from node i to node $n+1$ with cost $u_i(\cdot)$ and an edge $i \to i+1$ with cost $o_i(\cdot)$, for every $i \in [n]$. Finally, there is another edge from n to $n+1$ with cost $o_n(\cdot)$. Finally, each node i has a supply of Δ_i and node $n+1$ is a sink with demand $\sum_i \Delta_i$. Given a feasible flow in this graph, the flow on edge $(i \to n+1)$ gives the time allocation for job i, and vice versa. Now, if the costs are piece-wise linear and convex, we have from the algorithm by Pinto and Shamir [22] for min flows with convex piece-wise linear costs, that the problem can be solved in polynomial (in n and the maximum number of pieces in the cost functions) time. The optimal solution to this problem is a 2-approximation, and the analysis closely resembles the proof of Theorem 3 (details deferred to full version).

3.2 Optimal Solution to RAS for Special Cases

Let us now investigate additional properties with the goal of strengthening our LP. We start by proving our claim from earlier – all jobs underaged (taking 0 time) and all jobs overaged (taking maximum time) are worst-cases for optimal allocations. We break down the proof into smaller parts. The first lemma is very useful and appears often in proofs of other lemmas. The key insight behind the lemma is simple – when all jobs overaged is a worst-case, if the first job is forced to start late, then the case of all jobs overaged suffers maximum increase in cost, and is thus still a worst case.

▶ **Lemma 4.** *Given an arbitrary allocation $\{A_i\}_i$, where $t_i = \Delta_i$ for all i is a worst-case. Recall that w.l.o.g., $A_1 = 0$ and consider a modified problem for the adversary, where the first job is always forced to start at a later time t_0 instead of time $A_1 = 0$. Then, $T = (t_i)$ is also a worst-case for the modified problem.*

Proof. Let $T = (\Delta_i)$ denote the profile for all jobs taking maximum time. When job 1 starts at time 0, denote the cost of job i by $c(0, T, i)$. Since T is a worst-case by assumption, we have $c(0, T) \geq c(0, Z)$ for every profile Z. For the modified setting where job 1 starts at time $t > 0$, we claim that $c(t, Z, i) \leq c(0, Z, i) + o_i t$ for every Z. To see this, suppose job 1 starts at 0 and consider two cases: (i) i overaged in Z and (ii) i underaged in Z. In case (i), i will still be overaged when job 1 starts at time t and the completion time of i can increase by at most t. In case (ii), if i is still underaged when job 1 starts at time t, we are done. Else, i becomes overaged but the maximum overage cost is $o_i t$. Now for profile T, since all jobs are overaged when job 1 starts at time 0, $c(t, T, i) = c(0, T, i) + o_i t$. Therefore, $c(t, Z) \leq \sum_i (c(0, Z, i) + o_i t) \leq \sum_i c(0, T, i) + \sum_i o_i t = c(t, T)$. ◀

The next lemma says that we cannot have an optimal allocation where a certain job is underaged (overaged) in all the worst-case time profiles. Observe that if job i is always underaged, simply reducing the allocation x_i would give a strictly better allocation, contradicting optimality. Indeed, Lemmas 12 and 15 in [20] argue exactly this. However, that argument fails if there exists a worst-case where i is underaged with zero cost (occurs when $i - 1$ ends at A_{i+1} and $t_i = 0$, which we defined as a case of underage in Section 2). This gives rise to a subtle issue that demands a more involved argument (similarly for the case of overage). We postpone the formal proof to the full version.

▶ **Lemma 5.** *Given an optimal allocation, consider an arbitrary job i. There exists a worst-case $T = (t_i)$ where i is underaged and $t_i = 0$, as well as a worst-case where i is overaged (with some t_i that is not necessarily Δ_i).*

▶ **Lemma 6.** *In every optimal allocation, the case of all jobs underaged, i.e., $t_i = 0$ for every job i, is a worst-case.*

Proof. Let $\{A_i\}_i$ denote an optimal allocation. Lemma 5 implies there is a worst-case where job 1 takes zero time. We proceed via induction, assuming there is a worst-case $T = (t_i)$ where $t_i = 0$ for $i \in [k] = \{1, \ldots, k\}$, i.e., all jobs in $[k]$ are underaged. We will show there exists a worst-case where all jobs in $[k+1]$ are underaged.

Suppose that $k+1$ is overaged in T (otherwise we are done). Lemma 5 implies there is a worst-case, denoted $T' = (t'_i)$, where job $k+1$ is underaged and takes zero time. Let C_{k+1} denote the completion time of job $k+1$ in T. Clearly, $C_{k+1} > A_{k+2}$ and $k+1$ starts at A_{k+1} in T. Similarly, let C'_{k+1} denote the start/completion time of job $k+1$ in T'. Then, $C'_{k+1} \leq A_{k+2}$. Now consider a new profile $Z = (z_i)$, formed by a combination of T and T'. We let $z_i = t'_i$ for $i \in [k]$ and $z_i = t_i$ for $i \in \{k+2, \ldots, n\}$. Since the completion times of job k are identical in T' and Z, we set $z_{k+1} = C_{k+1} - C'_{k+1} \leq C_{k+1} - A_{k+1} = t_{k+1}$. Therefore, job $k+1$ ends at time C_{k+1} in Z. Now, observe that $\sum_{i \in [n]} c(Z, i) = \sum_{i=1}^{k} c(T', i) + \sum_{i=k+1}^{n} c(T, i)$. Since T is a worst-case, we also have $\sum_{i=1}^{k} c(T', i) + \sum_{i=k+1}^{n} c(T, i) \leq \sum_{i=1}^{n} c(T, i)$. Therefore, $\sum_{i=1}^{k} c(T', i) \leq \sum_{i=1}^{k} c(T, i)$. Now, consider another hybrid case Q, where jobs 1 to k are all underaged and take zero time as in T and jobs $k+2$ to n are as in T'. Job $k+1$ starts/ends at A_{k+1} in Q and hence $c(T', k+1) \leq c(Q, k+1)$. Combining everything, $\sum_i c(T', i) \leq \sum_{i=1}^{k} c(T, i) + \sum_{k+1}^{n} c(T', i) \leq \sum_{i=1}^{k} c(T, i) + c(Q, k+1) + \sum_{k+2}^{n} c(T', i) = \sum_i c(Q, i)$. Hence, Q is a worst-case with jobs 1 to $k+1$ all underaged. ◀

▶ **Lemma 7.** *In every optimal allocation, the case of all jobs overaged is a worst-case. Therefore, $t_i = \Delta_i$ for all i is a worst-case.*

Proof. Let $\{A_i\}_i$ denote an optimal allocation. Observe that if there exists a worst-case where jobs $S_k = \{k, \ldots, n\}$ are all overaged for some k, then there exists a worst-case where all jobs in S_k are overaged and take maximum time. We proceed by induction and show that if there is a worst-case where every job in S_k is overaged, then there exists a worst-case where every job in S_{k-1} is overaged. For $k = n$, by Lemma 5 there is a worst-case where job n is overaged. Assume there exists a worst-case where jobs k to n are overaged, denoted as $T = (t_i)$.

Suppose that job $k-1$ is underaged in profile T (otherwise we are done). Recall that $u_{k-1} > 0$ and $t_{k-1} = 0$, since k is underaged in T. Then, since T is a worst-case profile; restricted to the subset S_k, the profile $\{t_i = \Delta_i\}_{i \geq k}$ is a worst-case with all jobs overaged, for allocation $\{A_i\}_{i \geq k}$. Now, by Lemma 5 for the set of all jobs $[n]$, there exists a worst-case where job $k-1$ is overaged, denoted $T' = (t'_i)$. Then, consider profile $Z = (z_i)$ with $z_i = t'_i$ for jobs in $S_1 - S_{k-1}$ and $z_i = \Delta_i$ for $i \in S_{k-1}$. Z is a worst-case profile since it matches the cost in T' for jobs in $S_1 - S_{k-1}$, and due to Lemma 4 the total costs for jobs in S_{k-1} can only be larger than the same total in T'. ◀

The following equation characterizes the worst-case cost of every optimal allocation $\{x_i\}_i$,

$$\sum_{i=1}^{n} o(i)(\Delta_i - x_i) = \sum_{i=1}^{n} u_i x_i. \tag{3}$$

We can already modify LP-1 by adding the equality above. However, this property alone is not sufficient to offer improved results even for identical underage costs. To that end, we introduce the following technical property.

▶ **Lemma 8.** *Given an optimal allocation $\{x_i\}_{i \in [n]}$,*

$$\sum_{j=k}^{n} (u_j + o(j)) y_j \geq \sum_{j=k}^{n} o(j) \Delta_j \quad \textit{for all } k \in [n]. \tag{4}$$

To get some intuition behind the lemma, consider the objective of minimizing $\sum_i u_i y_i$ for non-negative y_i subject to the equation (3). A greedy solution that sets $y_i = \frac{\sum_j \Delta_j o(j)}{u_i + o(i)}$ for $i = \arg\min_{j \in [n]} \frac{u_j}{u_j + o(j)}$ is optimal. Here the job with the minimum ratio $\frac{u_i}{u_i + o(i)}$, bears the entire burden of the equality (3). Lemma 8 says that this can only occur when $i = n$ and more generally, places a lower bound on the contribution to equality (3) from values y_k to y_n, for all k. The proof of the lemma is rather technical and is postponed to the full version, along with two accompanying helper lemmas. Consider now the strengthened formulation,

$$\textbf{LP-2:} \quad \min \sum_{j=1}^{n} u_j y_j$$

$$\text{s.t.} \quad \sum_{j=1}^{n}(u_j + o(j))y_j = \sum_{j=1}^{n} o(j)\Delta_j \tag{5}$$

$$\sum_{j=k}^{n}(u_j + o(j))y_j \geq \sum_{j=k}^{n} o(j)\Delta_j \quad \forall k \in [n] \tag{6}$$

$$\sum_{j=1}^{k}(y_j - \Delta_j) \leq 0 \quad \forall k \in [n]$$

$$y_j \geq 0 \quad \forall j \in [n]$$

Clearly, every optimal allocation is a feasible solution for the LP. However, an optimal solution to the LP need not have all jobs underaged (or overaged) as worst-case, and hence need not be an optimal allocation. This is demonstrated by the example used in Theorem 3, where it is easily checked that the additional constraints given by (5) and (6) do not improve the worst-case approximation bound from earlier. However, we show that under additional assumptions LP-2 yields optimal allocations.

The recipe behind proving this is as follows: (i) Using the assumptions, show that there is actually a closed-form optimal solution to the LP. (ii) Show that for this solution, all jobs underaged and all jobs overaged are worst-cases. In particular, using this recipe for the special case of homogeneous underage costs $u_i = u$ for every i, we find that $x_i = \frac{o(i)\Delta_i}{u+o(i)}$ for all $i \in [n]$ is an optimal solution to LP-2 and an optimal allocation. More generally, we have the following.

▶ **Theorem 9.** *If $u_i \leq u_{i+1}\frac{o(i)}{o(i+1)}$ for all $i \in [n-1]$, then the allocation given by $x_i = \frac{o(i)\Delta_i}{u+o(i)}$, $i \in [n]$, is an optimal allocation.*

This generalizes and offers a different perspective on Theorem 5 in [20] (since $u_i = u$ for every i, implies $u_i \leq u_{i+1}\frac{o(i)}{o(i+1)}$ for all $i \in [n-1]$). As a direct implication of the above, we have a closed form optimal allocation if if the underage costs are non-decreasing i.e.,

$$u_i \leq u_{i+1} \quad \forall i \in [n-1].$$

This includes for instance, the case of increasing underage costs where $u_{i+1} \geq u_i + o_i$. As we discussed earlier in Section 1, this complements the assumptions made in previous work, where the common assumption is that the underage costs are not increasing too drastically and in particular that, $u_{i+1} \leq u_i + o_i$ for all $i \in [n-1]$.

Next, we claim that even if $u_i > u_{i+1}\frac{o(i)}{o(i+1)}$ for some i, there may still exist an LP optimal solution that is also an optimal allocation. This generalizes Theorem 9. A direct corollary of the result is that if underage costs are “large enough”, we have a closed form optimal solution even if the costs otherwise vary arbitrarily. More specifically, if the underage costs are such that for ever pair of jobs $i - l$ and i with $u_{i-l} > u_i$, we have,

$$u_i \geq \sum_{j=1}^{l} o_{i-j}.$$

Then LP-2 leads to a closed form optimal allocation.

▶ **Theorem 10.** *Given n jobs with parameters such that for $i \in [n]$, whenever $i \neq m_i \coloneqq \min\left(\arg\min_{j \geq i} \frac{u_j}{u_j + o(j)}\right)$, we have $u_{m_i} \geq o(i) - o(m_i)$, then the following is both an optimal solution to the LP and an optimal allocation,*

$$x_k = \frac{\sum_{i|k=m_i} \Delta_i o(i)}{u_k + o(k)} \quad \text{for all } k \in [n].$$

Optimal Allocation Beyond Constant Per-unit Costs

For non-decreasing, piece-wise linear and convex costs $o_i(\cdot), u_i(\cdot)$, let $\overline{o}_i, \underline{o}_i$ denote the largest and smallest slope for $o_i(\cdot)$ and, $\overline{u}_i, \underline{u}_i$ the largest and smallest slope for $u_i(\cdot)$. By generalizing insights from the case of constant per-unit costs, we show that Algorithm 1 finds the optimal allocation when $\underline{u}_{i+1} \geq \overline{u}_i$, for all $i \in [n-1]$. This condition is satisfied for instance if $u_i(x) = u \cdot x$ for all $i \in [n]$, and $o_i(\cdot)$ is an arbitrary non decreasing, convex piece-wise linear function. Similar to the case of constant per-unit costs, our assumption complements assumptions made in previous work on the stochastic setting of the problem. For instance, in the stochastic setting Ge et al. [10], gave a polynomial time algorithm for non decreasing, piece-wise linear and convex costs when $\overline{u}_{i+1} \leq \underline{u}_i + \underline{o}_i$, for all $i \in [n-1]$.

Algorithm 1 Allocation for Non-Linear Costs.

1: **for** $i = n$ to 1 **do**
2: $\quad \hat{o}_i(x) \coloneqq \sum_{j \geq i} \left(o_j(\Delta_i - x + \sum_{k|i<k\leq j}(\Delta_k - x_k)) - o_j(\sum_{k|i<k\leq j}(\Delta_k - x_k)) \right)$
3: $\quad x_i = \underset{x \geq 0}{\operatorname{argmin}} \max\{u_i(x), o_i(\Delta_i - x) + \hat{o}_i(x)\}$
4: **Output:** $\{x_i\}_i$

▶ Remark. Given two sets of jobs $S_{k-1} = \{k-1, \ldots, n\}$ and $S_k = \{k, \ldots, n\}$. Let the output of Algorithm 1 over set S_{k-1} be $\{x_i^{k-1}\}$, and over set S_k be $\{x_i^k\}$. Then we have, $x_i^{k-1} = x_i^k$ for all $i \in S_k$. Also, note that the algorithm takes n iterations, each involves a minimization of the maximum of two convex piece-wise linear functions (as $u_i(x)$ and $o_i(\Delta_i - x) + \hat{o}_i(x)$ are both piece-wise linear and convex in x). Therefore, each step involves finding the minimum of a convex piece-wise linear function. Hence, the algorithm runs in polynomial time (in n and the number of pieces in the cost functions).

4 Robust Appointment Scheduling and Ordering

So far, we assumed that jobs are given to us in fixed order and our task was to find an optimal appointment schedule. Focusing on the case of constant per-unit costs, we now consider the joint problem of finding an ordering and an appointment schedule such that the resulting cost is minimized. More formally, consider a permutation π over the set $[n]$, that determines the order of appointments. So given an ordering π, job i is the $\pi(i)$-th appointment in the schedule. We let A_i, C_i denote the start time and the completion time of the i-th appointment (or the $\pi^{-1}(i)$-th job). The joint scheduling and ordering problem can now be stated as,

$$\min_{\pi:[n]\to[n], \{A_i\}_i} \quad \max_{t_i \in [0,\Delta_i]\, \forall i} \sum_{i=1}^{n} \max\{o_{\pi^{-1}(i)}(C_i - A_{i+1}), u_{\pi^{-1}(i)}(A_{i+1} - C_i)\}. \tag{7}$$

Recall that under homogeneous underage costs $u_i = u \, \forall i$, we have a closed form solution for the problem for fixed permutation π. Letting $o(i) = \sum_{j|\pi(j)\geq\pi(i)} o_j$, the objective in this special case can be more simply stated as,

$$\min_{\pi:[n]\to[n]} \sum_{i=1}^{n} \frac{\Delta_i o(i) u}{o(i)+u}.$$

Let us call this problem RASO-H for brevity. Mittal et al. [20] showed that RASO-H reduces to an instance of min-sum scheduling with concave costs $1||\sum w_j f(C_j)$. Here given jobs j with processing time p_j, weight w_j, and a concave function $f(\cdot)$ over completion times C_j, the goal is to find an ordering that achieves the following,

$$\min_{\{C_j\}_j} \sum_{j=1}^{n} w_j f(C_j). \tag{8}$$

Indeed, letting $w_j = \Delta_j$, $p_j = o_j$ and $f(C_j) = \frac{uC_j}{C_j+u}$, we see that any ordering π for RASO-H is equivalent to an order π' in, $1||\sum w_j f(C_j)$. Here $\pi'(j) = n - \pi(j) + 1$ (i.e., orders are reversed as we move between the two problems). This reduces RASO-H to an instance of $1||\sum w_j f(C_j)$, while preserving the objective value. Therefore, algorithms for $1||\sum w_j f(C_j)$ can be used directly for RASO-H without any loss in guarantee. Using a result in [12], the following rule,

Smith's rule: Schedule jobs in the order of descending ratios $\frac{w_j}{p_j}$ (or ascending ratios $\frac{\Delta_j}{o_j}$),

is 1.137 approximate for RASO-H. In fact, there is an EPTAS for RASO-H due to the EPTAS for $1||\sum w_j f(C_j)$ [25]. Howevever, the hardness of RASO with a single underage cost, and more generally of $1||\sum w_j f(C_j)$, remains an intriguing open problem in scheduling theory [25, 19, 12, 20].

In the upcoming section, we consider the general problem for which no results were previously known. We give evidence indicating that the problem becomes much harder without the homogeneous underage costs assumption.

4.1 RASO with General Underage Costs

We show that RASO is strongly NP-hard when there are at least two different underage costs via a reduction from the strongly NP hard problem of min-sum scheduling on identical parallel machines, $P||\sum w_j C_j$ (problem SS13 in Garey and Johnson [9]). Further, we also show that no list ordering rule (such as Smith's rule) can be better than $O(n)$ approximate for the problem, ruling out the existence of "simple" approximation heuristics. We defer the details and proofs to the full version.

To develop approximation heuristics for RASO, unlike RASO-H, we cannot rely on a closed-form solution for the optimal allocation problem to simplify the problem. However, we do have a closed-form solution with cost guaranteed to be within a constant factor of the optimal due to Theorem 3. This does not immediately lead to a tractable problem, and taking a different approach we instead consider the closed form allocation given by the formula $x_i = \frac{\Delta_i o(i)}{u_i+o(i)}$. Given order π, let $o(i) = \sum_{j|\pi(j)\geq\pi(i)} o_j$. For this allocation rule the ordering problem simplifies to,

$$\min_{\pi:[n]\to[n]} \sum_{j=1}^{n} \frac{\Delta_j o(j) u_j}{o(j)+u_j}. \tag{9}$$

This is equivalent to a min-sum scheduling problem of the form,

$$\min_{\{C_j\}_j} \sum_j f_j(C_j).$$

Here C_j is the completion time of job j which has processing time $p_j = o_j$. Functions $f_j(C_j) = \Delta_j u_j \frac{C_j}{C_j+u_j}$ are concave, but now we have a different function for each job. We show (details in full version) that given an α approximation for $1||\sum_j f_j(C_j)$, there is a $4\alpha n$ approximation for RASO. The additional factor of $4n$ arises from the fact that (9) does not represent the true objective (7). Recall that the closed form allocation formula $x_j = \Delta_j \frac{o(j)}{o(j)+u_j}$, is not only suboptimal but can be arbitrarily worse than the optimal allocation for certain orders. So it is perhaps surprising that using this allocation we can still achieve some approximation bound. Note that for the min-sum scheduling problem, $1||\sum f_j(C_j)$, [7] gives a $4+\epsilon$ approximation for arbitrary f_j. Using this algorithm, we have a $(16+\epsilon)n$ approximation for RASO. We also show that no heuristic that orders jobs based on a simple ratio can beat $\Omega(n)$. Here we use the term "simple ratio" to refer to any real valued function evaluated independently for each job, using only the job parameters (o_i, Δ_i, u_i). A list ordering heuristic for this function then simply orders jobs in ascending or descending order of the function values.

4.2 Homogeneous Underage Costs

In this section, we develop some easy to implement heuristics with improved approximations for RASO-H. As we mentioned earlier, the computational complexity of RASO-H and more generally, $1||\sum w_j f(C_j)$ with concave cost function f, remains open but, there is a scaling based EPTAS for RASO-H due to the EPTAS for $1||\sum w_j f(C_j)$. The EPTAS is non-trivial and not easy to implement in practice. Practical heuristics such as Smith's rule, can be suboptimal for RASO-H even with just two jobs. This motivates us to consider the following heuristic,

Customized-Smith's rule (C-Smith): Schedule jobs in ascending order of $\frac{\Delta_i}{o_i(o_i+u)}$.

This straightforward heuristic is optimal for two jobs by design, and has an approximation guarantee of β, where $1.06036 < \beta < 1.06043$. While the order output by C-Smith is optimal for two jobs, we could more generally seek an order that is optimal w.r.t. exchanging the order of any two consecutive jobs. Let us call such an order *locally optimal*. The schedule output by both Smith and C-Smith is not locally optimal. In fact, no list-ordering heuristic can be locally optimal for all instances of RASO-H. To address this we introduce Algorithm 2, which outputs a locally optimal order. While not as simple as C-Smith, it is still fairly easy to implement – at each step, it computes a new set of ratios for the remaining jobs and picks the one with the best ratio, removing it from further consideration. However, a naive implementation has runtime $O(n^2)$, in contrast to $O(n \log n)$ for list ordering heuristics.

We show that Algorithm 2 outputs a solution that is at least as good as C-Smith on every instance. Thus, it is also β approximate. There exist instances for which the algorithm is not optimal. However, we leave finding the exact guarantee of the algorithm as an open problem.

▶ **Theorem 11.** *For every instance of RASO-H, Algorithm 2 outputs an order that is at least as good as the order given by C-Smith.*

The proof is deferred to the full version. It remains to show the β approximation for C-Smith. For this analysis it will be much more convenient both for clarity as well as notation, to focus on analyzing C-Smith for the scheduling problem $1||\sum_j w_j f(C_j)$, where

Algorithm 2 Locally Optimal Algorithm.

1: **Initialize:** $S = \emptyset, o(S) = 0$
2: **for** $i = 1$ to n **do**
3: Find,

$$j = \underset{k \in [n] \setminus S}{\operatorname{argmax}} \frac{\Delta_k}{o_k(o_k + o(S) + 1)}.$$

In case of a tie, pick the job with largest overage cost.

4: $\pi(j) = n - i + 1, S = S \cup \{j\}, o(S) = o(S) + o_j$
5: **Output:** $\pi(.)$

$f(C_j) = \frac{C_j u}{C_j + u}$ and $w_j = \Delta_j$,$p_j = o_j$. Note that we can let $u = 1$ w.l.o.g. Also recall, there is a cost preserving bijection between orders for $1||\sum_j w_j f(C_j)$ and RASO-H – reversing the order when moving from one to the other. Therefore, we will show the following equivalent theorem.

▶ **Theorem 12.** *For some constant $\beta \in [1.06036, 06043]$. scheduling jobs in descending order of $\frac{w_j}{p_j(p_j+1)}$ is exactly β approximate for the scheduling problem $1||\sum_j w_j \frac{C_j}{C_j+1}$.*

To prove the above theorem, we establish several intermediate results that characterize and simplify the worst-case instance for C-Smith. First, we show that there is a worst-case instance for C-Smith where all jobs are tied and in fact have ratio 1. This is shown via a generalization of Lemma 3.5 in [12] (Lemma 21 in [25]). Then, we show that in the worst-case C-Smith orders jobs in ascending order of processing times and the optimal order is the exact reverse. Interestingly, this is the opposite of Lemma 3.6 in [12] (Proposition 20 in [25]) for Smith's rule, where the optimal order is ascending in p_i and the worst order is descending. Given these properties, we can formulate a non-convex optimization problem in infinitely many variables, the optimal value of which is the approximation ratio, and every optimal solution is a worst-case instance. To then get lower and upper bounds on the optimum value, we utilize properties specific to our objective $f(x) = \frac{x}{x+1}$ to approximate the problem (which has infinitely many variables) with a family of optimization problems, that while still non-convex, have a finite number of variables. More complex objectives from the family give a tighter upper bound on true approximation ratio, but the number of variables involved increases. We find the global optimum to a problem in the family with five variables, and this closely matches our lower bound. We solve such a non-convex problem to global optimality by establishing upper and lower bounds on variables and using linear cuts, both of which allow us to then effectively use a nonlinear globally optimal MINLP solver, Couenne [13] (details in full version).

5 Conclusion & Open Problems

We considered the robust appointment scheduling problem with general underage costs. For the appointment scheduling problem with fixed jobs order, we found a simple LP that gives a 2-approximation for the problem under constant per-unit costs. Then we further refined this LP, resulting in a closed form solution for optimal allocations in special cases (generalizing previous results in the robust model, and complementing similar results in other models). We also showed that our results and approach extend more generally to convex piece-wise linear costs. When seeking an optimal allocation for the general case using our approach, more complications arise and it is not clear if one can still construct a linear (or convex) program such that an optimal solution to the program is also an optimal allocation. We leave finding an optimal solution for the general case (or showing hardness), as an open problem.

In the second setting, we considered the problem of jointly finding the optimal order and allocation given that order for the case of constant per-unit costs. For the case of heterogeneous underage costs, we show that the problem is strongly NP hard and no list ordering policy can do better than $\Omega(n)$ to approximate the optimal value. We also gave a heuristic that achieves this bound. Finding a better approximation for this setting remains another interesting open problem. For the case of homogeneous underage costs, we designed two simple and practical heuristics that are guaranteed to be with ≈ 1.06 of the optimal.

References

1 M. A Begen, R. Levi, and M. Queyranne. A sampling-based approach to appointment scheduling. *Operations Research*, 60(3):675–681, 2012.

2 M. A Begen and M. Queyranne. Appointment scheduling with discrete random durations. *Mathematics of Operations Research*, 36(2):240–257, 2011.

3 A. Ben-Tal and A. Nemirovski. Robust optimization–methodology and applications. *Mathematical Programming*, 92(3):453–480, 2002.

4 D. Bertsimas, D. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM Review*, 53(3):464–501, 2011.

5 D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

6 T. Cayirli and E. Veral. Outpatient scheduling in health care: a review of literature. *Production and Operations Management*, 12(4):519–549, 2003.

7 M. Cheung, J. Mestre, D. B. Shmoys, and J. Verschae. A Primal-Dual Approximation Algorithm for Min-Sum Single-Machine Scheduling Problems. *SIAM Journal on Discrete Mathematics*, 31(2):825–838, 2017.

8 B. Denton and D. Gupta. A sequential bounding approach for optimal appointment scheduling. *IIE Transactions*, 35(11):1003–1016, 2003.

9 M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

10 D. Ge, G. Wan, Z. Wang, and J. Zhang. A note on appointment scheduling with piecewise linear cost functions. *Mathematics of Operations Research*, 39(4):1244–1251, 2013.

11 L. V. Green, S. Savin, and B. Wang. Managing patient service in a diagnostic medical facility. *Operations Research*, 54(1):11–25, 2006.

12 W. Höhn and T. Jacobs. On the performance of Smith's rule in single-machine scheduling with nonlinear cost. *ACM Transactions on Algorithms*, 11(4):25, 2015.

13 IBM and Carnegie Mellon University. Couenne, an exact solver for nonconvex MINLPs. `https://projects.coin-or.org/Couenne/`, 2006.

14 R. Jiang, S. Shen, and Y. Zhang. Integer programming approaches for appointment scheduling with random no-shows and service durations. *Operations Research*, 65(6):1638–1656, 2017.

15 G. C. Kaandorp and G. Koole. Optimal outpatient appointment scheduling. *Health Care Management Science*, 10(3):217–229, 2007.

16 Q. Kong, C. Lee, C. Teo, and Z. Zheng. Scheduling arrivals to a stochastic service delivery system using copositive cones. *Operations Research*, 61(3):711–726, 2013.

17 A. Macario. Is it possible to predict how long a surgery will last? *Medscape Anesthesiology*, 108(3):681–685, 2010.

18 H. Mak, Y. Rong, and J. Zhang. Appointment scheduling with limited distributional information. *Management Science*, 61(2):316–334, 2014.

19 N. Megow and J. Verschae. Dual techniques for scheduling on a machine with varying speed. In *Automata, Languages, and Programming - 40th International Colloquium (ICALP)*, pages 745–756, 2013.

20 S. Mittal, A. S. Schulz, and S. Stiller. Robust appointment scheduling. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.

21 J. Orlin. Personal communication, 2018.

22 Y. Pinto and R. Shamir. Efficient algorithms for minimum-cost flow problems with piecewise-linear convex costs. *Algorithmica*, 11(3):256–277, 1994.

23 L. W. Robinson and R. R. Chen. Scheduling doctors' appointments: optimal and empirically-based heuristic policies. *IIE Transactions*, 35(3):295–307, 2003.

24 W. E. Smith. Various optimizers for single-stage production. *Naval Research Logistics*, 3(1-2):59–66, 1956.

25 S. Stiller and A. Wiese. Increasing Speed Scheduling and Flow Scheduling. In *Algorithms and Computation - 21st International Symposium (ISAAC)*, pages 279–290, 2010.

26 P. P. Wang. Static and dynamic scheduling of customer arrivals to a single-server system. *Naval Research Logistics*, 40(3):345–360, 1993.

27 P. P. Wang. Sequencing and scheduling n customers for a stochastic server. *European Journal of Operational Research*, 119(3):729–738, 1999.