Robust Appointment Scheduling

Shashi Mittal[1], Andreas S. Schulz[2], and Sebastian Stiller[3]

1 Amazon.com, Inc., Seattle, WA, USA
mshashi@alum.mit.edu
2 Massachusetts Institute of Technology, Cambridge, MA, USA
schulz@mit.edu
3 TU Berlin, Berlin, Germany
stiller@math.tu-berlin.de

Abstract

Health care providers are under tremendous pressure to reduce costs and increase quality of their services. It has long been recognized that well-designed appointment systems have the potential to improve utilization of expensive personnel and medical equipment and to reduce waiting times for patients. In a widely influential survey on outpatient scheduling, Cayirli and Veral (2003) concluded that the "biggest challenge for future research will be to develop easy-to-use heuristics." We analyze the appointment scheduling problem from a robust-optimization perspective, and we establish the existence of a closed-form optimal solution–arguably the simplest and best 'heuristic' possible. In case the order of patients is changeable, the robust optimization approach yields a novel formulation of the appointment scheduling problem as that of minimizing a concave function over a supermodular polyhedron. We devise the first constant-factor approximation algorithm for this case.

1998 ACM Subject Classification G.2.1 Combinatorics – Combinatorial Algorithms, G.1.6 Optimization – Nonlinear programming, I.1.2 Algorithms – Analysis of Algorithms

Keywords and phrases Robust Optimization, Health Care Scheduling, Approximation Algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.356

1 Introduction

We study the problem of appointment scheduling in a robust optimization framework. The appointment scheduling problem arises in many service operations where customers are served sequentially, service times of customers are uncertain, and one needs to assign time slots for serving the customers in advance. Arguably the most relevant practical setting where this problem arises is in health care services. Modern health care involves the usage of several high-cost devices and facilities such as MRI installations, CT scanners and operation rooms, in addition to highly trained and well-paid personnel. For health care providers, appointment scheduling is vital to ensure a high utilization of their resources as well as a high quality of service.[1] For example, consider the problem of scheduling surgeries for outpatients in an operation room at a hospital. The information about which surgeries are to be performed on a particular day is known in advance. However, the time needed to perform each surgery can vary. Typically on the preceding day, the hospital manager needs to decide the time at which a particular surgery is scheduled to start, and how much time to

[1] Excessive waiting times are often the major reason for patients' dissatisfaction in outpatient services. Nowadays, reasonable waiting times are expected in addition to clinical competence [5].

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 356–370
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

assign to that surgery. If the manager assigns a rather small time interval for a surgery, then it is more likely that the actual time of the surgery will exceed its assigned duration, thus delaying the next surgery. The inconvenience and costs resulting from the delay of both the patients and the staff constitute the *overage* cost of that surgery. If on the other hand, the hospital manager assigns an excessively long interval to a particular surgery, then chances are that the surgery may end early and the operation room will be left idle until the next surgery commences. In that case, the hospital incurs *underage* cost, which corresponds to the under-utilization of the resources in the operation room. Therefore, an ideal appointment schedule should achieve the right trade-off between the underage and the overage costs.

Existing models in the literature for the appointment scheduling problem include queueing models [20, 21], continuous stochastic models [6, 13, 17] and discrete stochastic models [1, 2]. In the stochastic models, the processing times of the jobs[2] are assumed to be independent random variables, and the objective is to find an appointment schedule that minimizes the expected cost. In all these models, one assumes complete knowledge about the distribution of the processing times of the jobs. However, in many service settings the distributions may not be known accurately, limiting the utility of the stochastic models. There might not be sufficient historical data of the processing times of the jobs to get a reasonable estimate of the probability distributions. Furthermore, because the cost function in the stochastic model is the expectation of a non-linear function of several random variables, the computational cost of finding an optimal schedule is significantly high. As a consequence, the methods employed to solve the problem are usually based on heuristics with no provable bounds on the running time of the algorithm or on the quality of the solutions they generate. Other methods require the use of advanced techniques such as Monte-Carlo simulations or submodular function minimization. Such techniques may not necessarily be practical in many situations. The drawbacks of the stochastic models mentioned above are not limited to the appointment scheduling problem, but are encountered in many situations in which there is uncertainty. Robust optimization is an alternative framework to address uncertainty. In robust optimization, the uncertainty in the input parameters is handled using uncertainty sets instead of random variables (see e. g. [3, 4]). Robust optimization models are often more tractable when compared to the corresponding stochastic optimization models.

Robust optimization models appear to be particularly useful in health care, as they attempt to reduce the cost and stress level of a bad day, rather than taking into account only the average. For example, stochastic models often give a "dome shaped" schedule, reckoning on the small probability of delays accumulating in the morning. But in case of several early delays such a schedule incurs a high cost as these delays spill over on a large part of the day. A robust schedule is not "dome shaped," but assigns some slack time in the morning as well.

The main contributions of this paper can be summarized as follows.

Robust Formulation of the Problem. We propose to look at the appointment scheduling problem in a robust optimization framework. For each job we only need the following information: the minimum and the maximum possible time the job will take to complete, the underage cost if the job finishes early, and the overage cost if the job finishes late. The objective in the robust model is to find a schedule for which the cost in the worst-case scenario of the realized processing times of the jobs is minimized. For simplicity of presentation, we

[2] From now on, we use the term 'job' to refer to the kind of service provided in any one specific context; for instance, in the health care setting, a job could correspond to a surgery.

focus on the important case where all underage cost coefficients are equal, i. e., they represent the opportunity cost of the facility.

A Closed-form Solution. We propose an intuitive method for scheduling jobs that aims to balance the maximum possible underage cost of a job with the maximum possible overage cost due to that job. This approach yields an optimal solution to the robust model. Its biggest advantage is that it gives a simple, easy to compute, closed-form solution for the optimal duration assigned to each job. Unlike existing, stochastic methods for appointment scheduling, our solution is simple enough that it can be implemented in a spreadsheet, thus greatly enhancing its practical value.

Ordering Problem. In health care, one finds both, applications in which the ordering is fixed, and applications where reshuffling jobs is possible to further reduce the cost [5]. Despite the relevance of the case where reordering jobs is possible, most research on methods to solve appointment scheduling neglect the optimization potential of ordering. One reason may be that most methods are already involved even without considering the ordering problem. We show that the closed-form expression for the optimal solution for the fixed-ordering problem can be used to derive constant-factor approximation algorithms for the ordering problem.

Related Work

An overview of the appointment scheduling problem is given in [5]. The existing literature on appointment scheduling can roughly be divided into three categories: queueing models, stochastic optimization models and stochastic models which use notions of discrete convexity, for example, submodular functions over an integer lattice. We discuss the relevant literature for all three models below.

Wang [20] proposes a queueing model for the problem, in which the processing times of the jobs are assumed to be independent and identically distributed random variables with exponential distribution. Both static and dynamic problems (i. e. the case when all the information about the jobs is not known in advance) are considered in this model, and an optimal schedule is obtained by solving a set of non-linear equations. In [21], the model is generalized to the case where the jobs can have different mean processing times. For this model, it is shown that the optimal sequence of the execution of the jobs is to process them in the increasing order of their mean processing times.

Denton and Gupta [6] formulate the problem as a two-stage stochastic linear program, and then use a sequential bounding algorithm to solve the corresponding stochastic optimization problem. They also give general upper bounds on the cost of a schedule which do not depend on the particular distribution of the processing times or the cost parameters of the jobs. Robinson and Chen [17] use a Monte Carlo integration technique to compute near-optimal solutions for the appointment scheduling problem. They show that an optimal schedule for this model has a “dome shaped” structure. That is, the allowances for the assigned durations for the jobs first increase, and then decrease steadily for jobs in the end of the sequence. They also give heuristics which approximate this dome shaped structure of the optimal schedule. Green et al. [8] consider the problem of outpatient appointment scheduling in which serving emergency patients is also permitted. They formulate the problem as a dynamic stochastic control problem and establish properties of an optimal policy for real-time scheduling and capacity allocation. Yet another way of computing an appointment schedule is using local search: Kandoorp and Koole [13] show that a local search algorithm converges to an optimal schedule. Gupta [9] considers the problem of optimally sequencing two jobs, and establishes the optimality of an ordering when a stochastic dominance condition holds for the distribution of the processing durations of the two jobs.

More recently, Begen and Queyranne [2] show that when the processing times of the jobs are discrete random variables with finite integer support, then there is an optimal schedule which is integral (i. e. the assigned starting times of the jobs have integer values in the optimal solution). They also show that under very general conditions, the cost function with respect to an integer appointment schedule is submodular. An optimal solution can then be found using well known algorithms for submodular function minimization (e. g. [12, 15]). This idea has also been extended to a get a near-optimal schedule for a data driven model [1], where the processing time distributions of the jobs are not known in advance, but instead one uses the past data of the realized processing times of the jobs to approximate the distributions.

2 Appointment Scheduling and Ordering

The appointment scheduling problem consists of determining starting times for a fixed sequence of jobs with uncertain processing times. When the appointment schedule is executed and the exact processing times materialize, each jobs starts at its planned appointment time or at the completion time of its predecessor, whichever comes later. The goal is to minimize a weighted sum of idle times of the server and delays of the jobs. In the scheduling and ordering version of the problem, one also decides on the order in which the jobs are processed.

Formally, instances of both the *robust appointment scheduling and ordering problem* and the *robust appointment scheduling problem* are given by an n-tuple of jobs and a positive value u, the cost per unit of idle time. Each job i is characterized by $[\underline{p}_i, \overline{p}_i]$, an interval of (non-negative) possible processing times, and a positive value o_i, the per unit-time cost for the delay of job i.

A solution to the robust appointment scheduling problem is a vector of appointments $A = (A_1, \ldots, A_{n+1})$. Here, $A_1 = 0$ is the scheduled start time of job 1, A_{n+1} is the planned completion time of job n, and A_i is the planned completion time of job $i-1$ (and, simultaneously, the planned start time of job i), for all $2 \leq i \leq n$. A solution (π, A) to the robust appointment scheduling and ordering problem is a permutation $\pi : [n] \to [n]$ of the jobs' indices, and an appointment vector for this permutation as above, i. e., $(A_{\pi^{-1}(1)}, \ldots, A_{\pi^{-1}(n+1)})$.[3]

A scenario is any vector $P = (p_1, \ldots, p_n)$ of processing times within the given intervals: $p_i \in [\underline{p}_i, \overline{p}_i]$. We call $\mathbb{P} = \prod_{i=1}^{n} [\underline{p}_i, \overline{p}_i]$ the scenario set. In the following definitions we focus on the scheduling and ordering problem. Fixing the permutation π to the identity (i. e., ignoring it in the formulas) gives the definitions for completion time and cost in the version of the problem without ordering.

For a solution (π, A), the completion time of job $\pi^{-1}(i)$ in a scenario P is

$$C_{\pi^{-1}(i)} = \max(A_{\pi^{-1}(i)}, C_{\pi^{-1}(i-1)}) + p_{\pi^{-1}(i)}.$$

The cost of a solution (π, A) in a scenario P is

$$F\big((\pi, A), P\big) = \sum_{i=1}^{n} \max\big(u(A_{\pi^{-1}(i+1)} - C_{\pi^{-1}(i)}), o_{\pi^{-1}(i)}(C_{\pi^{-1}(i)} - A_{\pi^{-1}(i+1)})\big).$$

[3] As usual, $[n]$ denotes the set $\{1, 2, \ldots, n\}$. In a slight abuse of notation, $A_{\pi^{-1}(n+1)}$ refers to the planned completion time of the last job.

We call the first argument of the maximum the *underage cost*, and the second the *overage cost*. We say a job is *underaged*, respectively, *overaged* depending on which of the two terms is positive.[4] The cost of a solution (π, A) is

$$F(\pi, A) := \sup_{P \in \mathbb{P}} F\big((\pi, A), P\big) = \max_{P \in \mathbb{P}} F\big((\pi, A), P\big),$$

where the second equality holds because F is continuous in P [2].

It is helpful to use $a_i := A_{i+1} - A_i$ as a short-hand notation for the time assigned to job i. We will sometimes identify the schedule A by the vector of assigned times $a = (a_1, \dots, a_n)$. For convenience, let us also introduce the following notation: $o_{\geq i} := \sum_{j \geq i} o_j$ and $\Delta_i := \overline{p}_i - \underline{p}_i$.

The robust appointment model allows for a number of structural and algorithmic insights. In this extended abstract, we focus on two main results:

▶ **Theorem 1.** *The robust appointment scheduling problem has a closed-form optimal solution.*

In addition to the obvious practical implications, Theorem 1 is also essential in deriving the next result.

▶ **Theorem 2.** *The robust appointment scheduling and ordering problem admits a polynomial-time approximation algorithm with constant-factor performance guarantee.*

We first design a $(2 + \epsilon)$-approximation algorithm that is of interest in its own right and provides very good insights into the essential structure of the robust appointment scheduling and ordering problem. We also give a polynomial-time approximation scheme, which is based on a new formulation of the robust appointment scheduling and ordering problem as that of minimizing a concave function over a supermodular polyhedron.

3 Robust Appointment Scheduling (with Fixed Job Order)

In this section, we consider the problem of finding optimal appointment times for the robust appointment scheduling problem when the job order is fixed to $1, 2, \dots, n$. We prove the optimality of an appointment schedule that is based on the following idea: charge each part of the total cost to the job that caused it. For the total underage cost, responsibilities are clear: in case $A_{i+1} - C_i > 0$, we want to charge job i for $u(A_{i+1} - C_i)$ of the total cost. Note that $A_{i+1} - C_i \leq a_i - \underline{p}_i$.

The overage cost of job i, i. e., $o_i(C_i - A_{i+1}) > 0$, can be caused by job i itself, namely in case $a_i < p_i$, and by jobs j preceding i that initially cause a delay, i. e., $a_j < p_j$, which spills over to i. We want to distribute the cost $o_i(C_i - A_{i+1})$ to those jobs that initially caused it. Note that $C_i - A_{i+1} \leq \sum_{j \leq i}(p_j - a_j)^+$ and, by reordering of summation,

$$\sum_i o_i \sum_{j \leq i} (p_j - a_j)^+ = \sum_j o_{\geq j}(p_j - a_j)^+ \leq \sum_{j \text{ overaged}} o_{\geq j}(\overline{p}_j - a_j).$$

Hence, we can distribute the entire cost of schedule A in any scenario P by attributing to each underaged job i at most $u(a_i - \underline{p}_i)$ of underage cost and to each overaged job i at most $o_{\geq i}(\overline{p}_i - a_i)$ of overage cost. Formally, we have

[4] In the model considered here, the cost caused by the overage of the i-th job in the sequence depends only on that job. All of our results remain to hold true if the overage cost of the i-th jobs is determined by the $(i + 1)$-st job in the sequence.

▶ **Lemma 3.** *For any appointment schedule A and any scenario $P \in \mathbb{P}$, the cost $F(A, P)$ is bounded as follows:*

$$F(A, P) \leq \sum_{i=1}^{n} \max\big(u(a_i - \underline{p}_i), o_{\geq i}(\overline{p}_i - a_i)\big).$$

Note that the expression on the right-hand side does not depend on P.

This is a very pessimistic perspective. Still, our solution is guided by this pessimism: Every job i shall minimize the maximum over its own maximal underage cost, namely $u(a_i - \underline{p}_i)$, and the maximum total increase in overage cost job i can cause: $o_{\geq i}(\overline{p}_i - a_i)$. Setting these two terms equal, i. e., balancing the potential underage cost and the potential responsibility for overage cost, yields the a_i of the *balanced schedule*.

▶ **Definition 4.** For an instance of the robust appointment scheduling problem, the *balanced schedule* A^B is defined by

$$a_i^B = \frac{u\underline{p}_i + o_{\geq i}\overline{p}_i}{u + o_{\geq i}}$$

for all $i = 1, \ldots, n$.

In this section we show that the *balanced schedule* is optimal and determine its cost, which gives a refined version of Theorem 1.

▶ **Theorem 5.** *For any instance of the robust appointment scheduling problem the corresponding balanced schedule A^B is an optimal solution. The cost of the balanced schedule A^B is*

$$F(A^B) = \sum_{i=1}^{n} \frac{uo_{\geq i}\Delta_i}{u + o_{\geq i}} \quad .$$

The pessimistic cost terms balanced in A^B are the actual costs of jobs in scenarios with the following properties: All jobs are either at maximal or minimal length, and if job i is at maximal length, then so are all jobs after job i. We prove Theorem 5 by showing that this set of $n + 1$ scenarios is the set of worst-case scenarios for any optimal solution.

In the rest of this section, we will prove Theorem 5. The following intuitive lemma helps to eliminate some pathological cases from further consideration.

▶ **Lemma 6.** *There exists an optimal appointment schedule A for which $\underline{p}_i \leq a_i \leq \overline{p}_i$.*

Proof. The first part of the inequality was proved in [2]. For the second part, suppose that in an optimal solution A, the duration assigned to some job i is greater than $\overline{p}_i$. Let i be the largest index of a job for which this is the case. Let $\delta = A_{i+1} - A_i - \overline{p}_i$. By assumption, $\delta > 0$. Note that we can focus on $i > 1$. Otherwise we could simply reduce each A_j by δ, for $j = 2, 3, \ldots, n$, and the resulting schedule would have lower cost, which is a contradiction to the optimality of A. We claim that changing A_i to $A_i + \delta$ does not increase the cost in any scenario. We consider two cases:
Case 1: Job $i - 1$ is underaged. In this situation, job i is underaged as well. If A_i is changed to $A_i + \delta$, then the underage cost of job $i - 1$ increases, but the underage cost of job i decreases by the same amount.
Case 2: Job $i - 1$ is overaged. Let C_{i-1} be the completion time of job $i - 1$. Then $C_{i-1} > A_i$. If $C_{i-1} - A_i \geq \delta$, then increasing A_i to $A_i + \delta$ only decreases the overage cost of job $i - 1$,

and changes nothing else. If $C_{i-1} - A_i < \delta$, then after increasing A_i to $A_i + \delta$, job $i-1$ becomes underaged. However, any increase in the underage cost of job $i-1$ is neutralized by the decrease in the underage cost of job i. The net effect is a decrease in the overall cost, as the overage cost that job $i-1$ was incurring in the earlier schedule has disappeared in the new schedule.

As a result, the cost in every scenario either remains the same, or decreases upon increasing A_i by δ. Therefore, the new schedule is still optimal, and the largest index of some job violating the claim of the lemma is now smaller than i. Iterating the argument, if necessary, completes the proof. ◀

From now on, we will focus on optimal appointment schedules that satisfy the condition in Lemma 6. Let $\mathcal{P}$ denote the set of all scenarios P such that $p_i \in \{\underline{p}_i, \overline{p}_i\}$ for all jobs i. We call these scenarios the *extremal scenarios.* We can show constructively that the cost of any scenario does not decrease by shifting it to an extremal scenario. This eventually shows that we can limit our attention to a finite scenario set.

▶ **Lemma 7.** *Let A^* be any schedule that has the lowest worst-case scenario cost, when only the scenarios in the set $\mathcal{P}$ are considered. Then A^* is an optimal appointment schedule.*

For the proof of Lemma 7 we need some preparation.

▶ **Definition 8.** For an appointment schedule A and a realization of the processing times of jobs P, a *chain* is defined as:

- A single job that is not overaged, or
- A sequence of jobs $i, i+1, \ldots, j$, such that jobs i to $j-1$ are overaged and job j is not overaged, or
- A sequence of jobs $i, i+1 \ldots, j$ all of which are overaged and job j is the last job.

Note that, given an appointment schedule A, the actual execution of the jobs for a given realization of the processing times of jobs P is a union of consecutive chains.

▶ **Lemma 9.** *For any given appointment schedule A, there exists a worst-case scenario P such that $p_i \in \{\underline{p}_i, \overline{p}_i\}$ for all jobs i.*

Proof. Let P be a worst-case scenario for the given appointment schedule A. Suppose P does not satisfy $p_i \in \{\underline{p}_i, \overline{p}_i\}$. Let i be the smallest job index for which this property is violated. We will convert P to a new scenario P', such that p'_i is either $\underline{p}_i$ or $\overline{p}_i$, and the cost corresponding to P', $F(A, P')$, is at least $F(A, P)$. For such a job i, there are the following cases to consider:
Case 1: Job i is not overaged. In this case, setting $p'_i = \underline{p}_i$ and $p'_j = p_j$ for all other jobs j results in increasing the cost of job i, and changes nothing else for other jobs. Thus $F(A, P') \geq F(A, P)$.
Case 2: Job i is overaged, and it is the last job. Then setting $p'_i = \overline{p}_i$ and $p'_j = p_j$ for all other jobs j increases the cost of the last job, and the cost of all other jobs remains the same. Thus $F(A, P') \geq F(A, P)$ in this case as well.
Case 3: Job i is overaged, and it is not the last job. Consider the chain which job i is part of in the realization P. Let $i, \ldots, j$ be the sequence of jobs including and following i in this chain. There are three sub-cases to consider:

- Job j is overaged. This means that the chain of which i is a part of is the last chain. In this case, all the jobs from job i onwards are overaged. Let $p'_i = \overline{p}_i$ and $p'_k = p_k$ for all other jobs k. Then the overage cost of all the jobs i and onwards is higher in P' than in P, while the cost of all the other jobs remains unchanged. Therefore $F(A, P') \geq F(A, P)$.

- Job j is underaged and $o_i + \ldots + o_{j-1} \leq u$. Suppose we reduce the processing time of job i by a sufficiently small amount ϵ that keeps the chain which i is a part of intact. Then the overage cost of jobs $i, \ldots, j-1$ decreases by $(o_i + \ldots + o_{j-1})\epsilon$, while the underage cost of job j increases by $u\epsilon$. Since $o_i + \ldots + o_{j-1} \leq u$, the cost in the modified schedule is at least as much as that in the original schedule. If ϵ can be chosen such that $p'_i = \underline{p}_i$, we are done. Otherwise, the chain that i was part of gets split into smaller chains. However, setting $p'_i = \underline{p}_i$ and $p'_k = p_k$ for all other jobs k still results in a realization P' for which $F(A, P') \geq F(A, P)$.
- Job j is underaged and $o_i + \ldots + o_{j-1} > u$. In this case, let $p'_i = \overline{p}_i$, and $p'_k = p_k$ for all other jobs k. It is possible that by doing so, the underage cost of some job $k \geq j$ decreases, and it may even get overaged. Since $o_i + \ldots + o_{j-1} > u$, it follows that $o_i + \ldots + o_{k-1} > u$ as well. So even though the underage cost of job k decreases, it is more than compensated by the increase in the overage cost of jobs $i, \ldots, k-1$. This holds true for any underaged job $k \geq j$. Therefore, the cost in the realization P' is at least as much as that in the realization P.

We can continue this process for each job that is not extremal, and eventually we will obtain a worst-case scenario in which each job is extremal, and whose realized cost is at least as much as that of the original scenario. Thus, for any appointment schedule A, there exists a worst-case scenario P such that $p_i \in \{\underline{p}_i, \overline{p}_i\}$ for all jobs i. ◀

Lemma 9 implies that in order to compute the worst-case scenario cost of an appointment schedule, it suffices to consider extremal scenarios only:

▶ **Lemma 10.** *For a given appointment schedule A, its cost $F(A)$ is given by*

$$F(A) = \max_{P \in \mathcal{P}} F(A, P).$$

In turn, Lemma 10 completes the proof of Lemma 7. Recall that Lemma 7 states that in order to compute an optimal appointment schedule, it suffices to consider the optimal solution over extremal scenarios.

Proof of Lemma 7. Let A^* be a schedule that has the lowest worst-case scenario cost, when only the scenarios in the set $\mathcal{P}$ are considered. Suppose A^* is not an optimal appointment schedule. Let $\hat{A}$ be an optimal appointment schedule. Lemma 10 implies that there exists $\hat{P} \in \mathcal{P}$ such that $F(\hat{A}) = F(\hat{A}, \hat{P})$. We have the following chain of inequalities:

$$\max_{P \in \mathcal{P}} F(\hat{A}, P) = F(\hat{A}, \hat{P}) = F(\hat{A}) < F(A^*) = \max_{P \in \mathcal{P}} F(A^*, P).$$

Thus, we get $\max_{P \in \mathcal{P}} F(\hat{A}, P) < \max_{P \in \mathcal{P}} F(A^*, P)$, which contradicts the optimality of A^* over the scenarios in the set $\mathcal{P}$. ◀

According to Lemma 7, it suffices to consider the 2^n extremal scenarios when one wants to compute an optimal appointment schedule. For a schedule A we denote by $\mathcal{W}_A$ the set of all extremal, worst-case scenarios. We will now determine the elements of $\mathcal{W}_A$ completely for optimal schedules A. The finiteness of the scenario set $\mathcal{W}_A$ allows for ϵ-shifting arguments that facilitate the derivation of the following two lemmata.

▶ **Lemma 11.** *For any appointment schedule A, if job i is tight or underaged (i. e. $C_i \leq A_{i+1}$) in a worst-case scenario P, then $p_i = \underline{p}_i$.*

Proof. If job i is tight or underaged in a worst-case scenario P and the job is not at minimal length, then reducing the length of i will give a higher cost, contradicting P being a worst-case scenario. ◀

▶ **Lemma 12.** *For an optimal appointment schedule A, and for each job i there is at least one $P \in \mathcal{W}_A$ where job i is not underaged, and at least one $P' \in \mathcal{W}_A$ where job i is not overaged.*

Proof. Suppose job i is underaged for all scenarios in $\mathcal{W}_A$. As $\mathcal{W}_A$ is finite and u is positive, there is $\epsilon > 0$ such that decreasing a_i by ϵ gives a better solution, which is a contradiction. The proof for the other case is similar. ◀

A slightly more involved proof, again by using shifting arguments that exploit the finiteness of the worst-case scenario set, gives the following lemma.

▶ **Lemma 13.** *Let A be an optimal appointment schedule. Then for all jobs i, $1 \le i \le n-1$, there is a worst-case scenario $P \in \mathcal{W}_A$ such that job i is not overaged and job $i+1$ is not underaged.*

Proof. Consider an arbitrary, but fixed job i, $1 \le i \le n-1$. By Lemma 12, there is at least one scenario in $\mathcal{W}_A$ in which i is not overaged. Suppose that all scenarios in $\mathcal{W}_A$ in which job i is not overaged, also have job $i+1$ underaged. Let c^* be the worst-case scenario cost of A. For a given $\epsilon > 0$, let A^ϵ be the appointment schedule in which $A^\epsilon_{i+1} = A_{i+1} + \epsilon$, and $A^\epsilon_j = A_j$ for all other jobs j. As $\mathcal{P}$ is finite, and as $F(A, P)$ is a continuous function in A, there exists $\epsilon > 0$ such that the schedule A^ϵ satisfies the following properties:

- For all $P \in \mathcal{W}_A$, $F(A^\epsilon, P) \le F(A, P)$.
- For all $P \in \mathcal{W}_A$ such that job i is overaged in schedule A, $F(A^\epsilon, P) < F(A, P)$.
- For all $P \in \mathcal{P} \setminus \mathcal{W}_A$, $F(A^\epsilon, P) < c^*$. In particular, the worst-case scenarios for the appointment schedule A^ϵ belong to the set $\mathcal{W}_A$.
- For the schedule A^ϵ, in no scenario from the set $\mathcal{P}$ does job i finish exactly at time A^ϵ_{i+1}.

Again, as $\mathcal{P}$ is finite and as the cost function is continuous with respect to the appointment schedule, there is $0 < \delta < \epsilon$ such that decreasing a^ϵ_i to $a^\epsilon_i - \delta$ lets no scenario in $\mathcal{P} \setminus \mathcal{W}_{A^\epsilon}$ reach c^*, but all scenarios in which job i is underaged for schedule A have a lower cost as compared to A^ϵ. Thus, the worst-case scenario cost of the appointment schedule $(a^\epsilon_1, \ldots, a^\epsilon_{i-1}, a^\epsilon_i - \delta, a^\epsilon_{i+1}, \ldots, a^\epsilon_n)$ is less than c^*, contradicting the optimality of the appointment schedule A. ◀

The next lemma establishes a crucial property of the worst-case scenarios of an optimal schedule: any overaged job is followed by an overaged job.

▶ **Lemma 14.** *For an optimal solution A, for every $P \in \mathcal{W}_A$, if job i, for some $1 \le i \le n-1$, is overaged, then so is job $i+1$.*

Proof. Assume to the contrary, for some i there is $P_1 \in \mathcal{W}_A$ with job i overaged but job $i+1$ underaged or tight. By Lemma 13, there is a worst-case scenario $P_2 \in \mathcal{W}_A$ with job i underaged or tight, and job $i+1$ tight or overaged. Split the cost of P_1 into $c(P_1) = c_{\le i}(P_1) + c_{\ge i+2}(P_1) + c_{i+1}(P_1)$, i.e., the cost of jobs up to and including job i, the cost of jobs $i+2$ or later, and the underage cost of job $i+1$. In case $i = n-1$, we set the term $c_{\ge i+2}(P_1)$ equal to zero. For P_2 we split $c(P_2) = c_{\le i}(P_2) + c_{\ge i+1}(P_2)$. As both are worst-case scenarios we have $c(P_1) = c(P_2)$.

Claim: $c_{\le i}(P_2) < c_{\le i}(P_1)$. Else, replace the jobs up to and including i in P_1 by the corresponding jobs in P_2. This *strictly* increases the underage cost of $i+1$, because i is overaged in P_1, and keeps $c_{\geq i+2}(P_1)$ as it is. This way we get a scenario with strictly higher cost, contradicting $P_1 \in \mathcal{W}_A$.

Because of the claim and $c(P_1) = c(P_2)$ we must have $c_{\geq i+1}(P_2) > c_{\geq i}(P_1) + c_{i+1}(P_1)$. Now, construct a scenario P' using p_j of P_1 for all jobs $1 \leq j \leq i$ and, in case $i \leq n-2$, using p_j of P_2 for all jobs j with $i+2 \leq j \leq n$. As $i+1$ must be shorter in P_1 than in P_2, we can adjust p_{i+1} such that its completion time is the same as in scenario P_2. Thereby, in case $i \leq i-2$, the realized starting time of $i+2$ in P' is that of $i+2$ in scenario P_2.

Now, P' has strictly higher cost than the worst-case scenarios P_1 and P_2, which contradicts the assumption that P_1 and P_2 are worst-case scenarios for the optimal schedule A. Hence the statement of the lemma holds. ◀

The following lemma is the overage analog of Lemma 11.

▶ **Lemma 15.** *If for some optimal solution A and some worst-case scenario $P \in \mathcal{W}_A$ job i is overaged, then $p_i = \overline{p}_i$.*

Proof. If i is overaged, all jobs $j \geq i$ are overaged as well, by Lemma 14. But then, if one could increase the length of job i, the cost of A would increase, which is not possible, because P is a worst-case scenario. Hence $p_i = \overline{p}_i$. ◀

The following lemma shows that many extremal scenarios are simultaneously worst case.

▶ **Lemma 16.** *There exists an optimal appointment schedule for which at least $n+1$ extremal scenarios are worst-case scenarios.*

Proof. Let A^* be an optimal solution of a given instance of the appointment scheduling problem. For each $P \in \mathcal{P}$, the cost of A^* in the scenario P, $F(A^*, P)$ is given by a linear function $f_P(A)$. Here A corresponds to the assigned duration to each job. Consider the following linear program:

$$\begin{array}{ll} \min & C \\ \text{s.t.} & f_P(A) \leq C, \text{ for all } P \in \mathcal{P}. \end{array}$$

Clearly, A^* is a feasible solution of this linear program. By Lemma 7, an optimal solution of this linear program yields an optimal solution of the appointment scheduling problem. The linear program has $n+1$ variables, therefore in an optimal basic feasible solution, at least $n+1$ constraints must be satisfied with equality. Hence there exists an optimal appointment schedule in which at least $n+1$ extremal scenarios are worst-case scenarios. ◀

The next theorem characterizes the set of extremal worst-case scenarios for an optimal appointment schedule.

▶ **Theorem 17.** *There exists an optimal appointment schedule A for which the set of extremal worst-case scenarios is given by*

$$\mathcal{W}_A = \{(\underline{p}_1, \ldots, \underline{p}_i, \overline{p}_{i+1}, \ldots, \overline{p}_n) \;:\; 1 \leq i < n\} \cup \{(\underline{p}_1, \ldots, \underline{p}_n), (\overline{p}_1, \ldots, \overline{p}_n)\}.$$

Proof. Follows immediately from Lemmas 14, 15 and 16. ◀

We are now ready to prove the main theorem of this section.

Proof of Theorem 5. Equating the cost of the worst-case scenarios in which $k-1$ respectively k is the last job at minimal length gives $(a_k - \underline{p}_k)u = (\overline{p}_k - a_k)\sum_{i=k}^n o_{\geq i}$ for all $1 < k \leq n$. Equating the scenario with all jobs maximal with the one where only the first is minimal, gives the equation also for $k = 1$. These n equations uniquely determine the optimal schedule as the *balanced schedule*, A^B. Direct calculation of the cost of A^B for any of these scenarios gives the result. ◀

4 Robust Appointment Scheduling and Ordering

Theorem 5 reduces the robust appointment scheduling and ordering problem to finding a permutation $\pi : [n] \to [n]$ that minimizes the cost $F(\pi)$ of the corresponding balanced schedule:

$$\min_\pi \sum_{i=1}^n \frac{u \sum_{\pi^{-1}(j) \geq \pi^{-1}(i)} o_{\pi^{-1}(j)} \Delta_{\pi^{-1}(i)}}{u + \sum_{\pi(j) \geq \pi^{-1}(i)} o_{\pi^{-1}(j)}} .$$

By scaling cost coefficients, we may assume $u = 1$. Using variables $\Theta_i := \Theta_i^\pi := \sum_{\pi^{-1}(j) \geq \pi^{-1}(i)} o_{\pi^{-1}(j)}$ and taking cues from single-machine scheduling with weighted-sum-of-completion-times objective [16], we can restate the ordering problem as that of minimizing a concave function over a supermodular polyhedron:

$$\min \quad \sum_{i=1}^n \frac{\Delta_i \Theta_i}{1 + \Theta_i} \tag{1a}$$

$$\text{s.t.} \quad \sum_{j \in S} o_j \Theta_j \geq \frac{1}{2}\Big(\sum_{j \in S} o_j\Big)^2 + \frac{1}{2}\sum_{j \in S} o_j^2 \quad \text{for all } S \subseteq N. \tag{1b}$$

Here we use N to denote the set $[n] = \{1, 2, \ldots, n\}$ of all jobs.

▶ **Lemma 18.** *Linear program 1 is an exact formulation of the ordering problem.*

Proof. The objective function (1a) is strictly concave, which implies that an optimal solution is attained at an extreme point of the polyhedron described by the inequalities (1b). This polyhedron is supermodular (i. e., the right-hand side of (1b) viewed as a function of $S \subseteq N$ is supermodular). In particular, all extreme points are of the form $(o_{\geq\sigma(1)}, \ldots, o_{\geq\sigma(n)})$, where $\sigma : N \to N$ is a permutation of jobs and $o_{\geq\sigma(i)} = \sum_{j \geq i} o_{\sigma(j)}$ (see, e. g., [7]). This means that an optimal ordering of the jobs can be found by solving for an optimal extreme point of this formulation. Furthermore, if $(\Theta_1^*, \ldots, \Theta_n^*)$ is an optimal extreme point, then an optimal ordering of the jobs is given by a permutation σ which satisfies $\Theta_{\sigma(1)}^* \geq \ldots \geq \Theta_{\sigma(n)}^*$. ◀

The first result of this section is an ad hoc $(2+\epsilon)$-approximation algorithm for the robust appointment scheduling and ordering problem. Observe that for $\Theta_i \leq 1$ we have $\Theta_i \leq 2\frac{\Theta_i}{1+\Theta_i}$, whereas for $\Theta_i \geq 1$ we have $1 \leq 2\frac{\Theta_i}{1+\Theta_i}$. In particular, this readily implies that if o_i happens to be larger than 1 for all jobs i, then so is Θ_i for any ordering, and any ordering is a 2-approximation. If, on the other hand, $\sum_{i=1}^n o_i \leq 1$, it is straightforward to show that ordering by Smith's rule [18], i.e, by non-decreasing ratios of $\frac{\Delta_i}{o_i}$, gives a 2-approximation. We will have more to say about this rule later. We first show that the following optimization problem, inspired by these two extremal cases, leads to a $(2+\epsilon)$-approximation in general:

$$\min_{\substack{S \subseteq N \\ \sum_{i \in S} o_i \leq 1}} \Big(\sum_{i \notin S} \Delta_i + \min_{\pi : S \to S} \sum_{i \in S} \Delta_i \Theta_i^\pi \Big) \tag{2}$$

Note that after choosing S in Problem (2), Smith's rule on the elements of S yields an optimal order Θ for the "inner" minimization problem. Moreover, even though we do not require $\sum_{i\in S} o_i + o_j > 1$ for all j not in S, any j for which this is not the case can be added to S without increasing the objective function value. Therefore, we assume, for optimal or approximate solutions, that S is maximal with respect to $\sum_{i\in S} o_i \le 1$.

A solution S of Problem (2) can be turned into an ordering π_S of all jobs in N as follows:

1. Schedule first the jobs not in S in arbitrary order.
2. Afterwards, schedule the jobs in S using Smith's rule.

We first prove a lemma that establishes the connection between an approximate solution to Problem (2) and an optimal solution of the ordering problem.

▶ **Lemma 19.** *Let S^α be an α-approximate solution for Problem (2), and let π_{S^α} be the sequence of jobs produced by the above algorithm, if called with input S^α. Then π_{S^α} is a 2α-approximate solution of the robust appointment scheduling and ordering problem.*

Proof. Let S' be an optimal solution to Problem (2), and let π^* be an optimal ordering. We construct a solution S^* for Problem (2) from π^* as the set of all jobs i for which $\Theta_i^* \le 1$.

$$\begin{aligned} F(\pi_{S^\alpha}) &= \sum_{i\in N} \frac{\Theta_i^\alpha \Delta_i}{1+\Theta_i^\alpha} = \\ \sum_{i\notin S^\alpha} \frac{\Theta_i^\alpha \Delta_i}{1+\Theta_i^\alpha} + \sum_{i\in S^\alpha} \frac{\Theta_i^\alpha \Delta_i}{1+\Theta_i^\alpha} &\le \sum_{i\notin S^\alpha} \Delta_i + \sum_{i\in S^\alpha} \Delta_i \Theta_i^\alpha \le \qquad (3) \\ \alpha\left(\sum_{i\notin S'} \Delta_i + \sum_{i\in S'} \Delta_i \Theta_i'\right) &\le \alpha\left(\sum_{i\notin S^*} \Delta_i + \sum_{i\in S^*} \Delta_i \Theta_i^*\right) \le \qquad (4) \\ 2\alpha\left(\sum_{i\notin S^*} \frac{\Theta_i^* \Delta_i}{1+\Theta_i^*} + \sum_{i\in S^*} \frac{\Theta_i^* \Delta_i}{1+\Theta_i^*}\right) &= 2\alpha \sum_{i\in N} \frac{\Theta_i^* \Delta_i}{1+\Theta_i^*} = 2\alpha F(\pi^*). \end{aligned}$$

Inequality (3) is trivial as $\Theta_i^\alpha \ge 0$. The inequality in changing to line (4) is given by the approximation factor provided by the antecedent of the lemma. The inequality in line (4) holds by the optimality of S'. The last inequality follows from the construction of S^* because we have $1 \le 2\frac{\Theta_i^*}{1+\Theta_i^*}$ for all i not in S^* and $\Theta_i^* \le 2\frac{\Theta_i^*}{1+\Theta_i^*}$ for all i in S^*. ◀

From Lemma 19, it follows that in order to obtain a 2α-approximation algorithm for the ordering problem, it suffices to obtain an α-approximation algorithm for Problem (2). We show that we can actually obtain an FPTAS for this optimization problem by re-casting it as a modified knapsack problem.

Without loss of generality we assume that the jobs are sorted in the order of Smith's rule, i. e., $\Delta_1/o_1 \le \ldots \le \Delta_n/o_n$. Assuming rational input we can find integers Q and q_i such that $o_i = q_i/Q$ for all i. Since the objective function in Problem (2) is linear, it suffices to consider the optimization problem in which the objective function is scaled by Q. Problem (2) can then be recast as follows:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} \Delta_i \sum_{j=i}^{n} q_j x_j + \sum_{i=1}^{n} Q\Delta_i(1-x_i) && (5) \\ \text{s.t.} \quad & \sum_{i=1}^{n} q_i x_i \le Q, && (6) \\ & x_i \in \{0,1\} \text{ for all } i = 1,\ldots,n. && (7) \end{aligned}$$

Let x' be an optimal solution to the modified knapsack problem given by (5)-(7). Then an optimal solution S' of Problem (2) can be obtained as $S' = \{i \in N : x'_i = 1\}$. This problem can be solved in pseudo-polynomial time using dynamic programming, as stated in the following lemma.

▶ **Lemma 20.** *The modified knapsack problem given by* (5)-(7) *can be solved in pseudo-polynomial time by dynamic programming.*

Proof. We reduce the problem of solving the modified knapsack problem to that of the shortest path problem in a graph. For a problem instance corresponding to (5)-(7), we construct the corresponding graph G as follows. The nodes in the graph are indexed by (i, v) for $1 \leq i \leq n+1$ and $0 \leq v \leq Q$, where i corresponds to a job and v corresponds to the total contribution to the overage cost based on whether we decide to select the jobs for set S or not. From each node (i, v), there are at most two outgoing arcs:

1. Arc to node $(i-1, v+q_i)$ of cost $\Delta_i(v+q_i)$, provided $v + q_i \leq Q$ (this corresponds to choosing job i for set S).
2. Arc to node $(i-1, v)$ of cost $\Delta_i Q$ (this corresponds to not choosing job i for the set S).

All the nodes of the form $(1, v)$ in the graph are connected to a terminal node T. The optimal solution of the modified problem corresponds to a shortest path in graph G from the node $(n+1, 0)$ to the node T. Since there are $\mathrm{O}(nQ)$ nodes and arcs in the graph, the shortest path problem (and hence the modified knapsack problem) can be solved in the same amount of time. ◀

Similar to the FPTAS for the knapsack problem [11], we can obtain an FPTAS for the modified knapsack problem.

▶ **Lemma 21.** *There exists an FPTAS for the modified knapsack problem given by* (5)-(7).

Proof. The objective function (5) can be written as $g(x) = \sum_{i=1}^{n} a_i x_i + \sum_{i=1}^{n} b_i(1-x_i)$, where $a_i = \Delta_i \sum_{j=1}^{i} q_j$ and $b_i = \Delta_i Q$ for $i = 1, \ldots, n$. Let $C = \max_i(a_i, b_i)$. Consider the modified objective function given by

$$g'(x) = \sum_{i=1}^{n} a'_i x_i + \sum_{i=1}^{n} b'_i(1-x_i), \tag{8}$$

where $a'_i = \lfloor a_i n/\epsilon C \rfloor$ and $b'_i = \lfloor b_i n/\epsilon C \rfloor$. In $g'(x)$, all the coefficients have value at most n/ϵ. Using the dynamic programming algorithm given in the proof of Lemma 20, the problem with the modified objective function can be solved in $O((n^2/\epsilon))$ time, which is polynomial in n and $1/\epsilon$. It remains to show that the optimal solution for the problem with objective function $g'(x)$ is a $(1+\epsilon)$-approximation to the optimal solution of the problem with the original objective function g.

Let S^* (resp. S') correspond to the set of all jobs for which the corresponding variable x_i is 1 in the optimal solution for the objective function $g(x)$ (resp. $g'(x)$). Then, an upper bound on the objective function value of the solution S' is given by

$$\begin{aligned} g(S') &= \sum_{i\in S'} a_i + \sum_{i\notin S'} b_i \\ &= \frac{\epsilon C}{n}\sum_{i\in S'} \frac{a_i n}{\epsilon C} + \frac{\epsilon C}{n}\sum_{i\notin S'} \frac{b_i n}{\epsilon C} \\ &\le \frac{\epsilon C}{n}\sum_{i\in S'}\left(\left\lfloor \frac{a_i n}{\epsilon C}\right\rfloor + 1\right) + \frac{\epsilon C}{n}\sum_{i\notin S'}\left(\left\lfloor \frac{b_i n}{\epsilon C}\right\rfloor + 1\right) \\ &\le \frac{\epsilon C}{n} g'(S') + \epsilon C \\ &\le \frac{\epsilon C}{n} g'(S^*) + \epsilon C, \end{aligned}$$

where the last inequality follows from the fact that S' is an optimal solution for the objective function g'. Further, a lower bound on the objective function value of the optimal solution S^* is as follows:

$$\begin{aligned} g(S^*) &= \sum_{i\in S^*} a_i + \sum_{i\notin S^*} b_i \\ &\ge \frac{\epsilon C}{n}\left(\sum_{i\in S^*}\left\lfloor \frac{a_i n}{\epsilon C}\right\rfloor + \sum_{i\notin S^*}\left\lfloor \frac{b_i n}{\epsilon C}\right\rfloor\right) \\ &= \frac{\epsilon C}{n} g'(S^*). \end{aligned}$$

Putting both inequalities together, it follows that $g(S') \le g(S^*) + \epsilon C \le (1+\epsilon)g(S^*)$, as $g(S^*) \ge C$. Therefore, S' is a $(1+\epsilon)$-approximate solution to the modified knapsack problem. ◀

Putting the pieces together gives the first constant-factor approximation algorithm for the ordering problem.

▶ **Theorem 22.** *There is a $(2+\epsilon)$-approximation algorithm for the* robust appointment scheduling and ordering problem.

Proof. Follows from Lemma 19 and Lemma 21. ◀

An advantage of the $(2+\epsilon)$-approximation algorithm, apart from being ad hoc and using problem-specific insights, is that the jobs in the “non-Smith" part of the sequence can be scheduled in arbitrary order, which is especially appealing from a practical point of view. Interestingly, Lemma 18 actually opens the door to the use of more general machinery developed for the family of single-machine scheduling problems $1||\sum w_j f(C_j)$. Indeed, Lemma 18 implies that the robust appointment scheduling and ordering problem is equivalent to a single-machine scheduling problem of this type, where f is increasing and concave. For arbitrary concave f, it was shown in [19] that Smith's rule yields an approximation guarantee of $(\sqrt{3}+1)/2$. With the help of the theory developed in [10], one can actually get a refined approximation factor of 1.137 for Smith's rule for the particular concave function encountered here. Moreover, it follows from recent work (see [19, 14]) on $1||\sum w_j f(C_j)$ with concave resp. non-decreasing functions f that there is in fact a PTAS for the robust appointment scheduling and ordering problem.

We leave the computational complexity of the robust appointment scheduling and ordering problem as an open problem, and note that it is also unknown whether minimizing (monotone) concave functions over supermodular polyhedra is NP-hard.

Acknowledgments. The authors thank Mehmet Begen, Diwakar Gupta, Jim Orlin, Larry Robinson, David Shmoys, and Jose Verschae for stimulating discussions.

References

1 Mehmet A. Begen, Retsef Levi, and M. Queyranne. A sampling-based approach to appointment scheduling. Technical report, Sauder School of Business, University of British Columbia, 2008. Working Paper.

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

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

4 Dimitris Bertsimas and Melvyn Sim. The price of robustness. *Operations Research*, 52:35–53, 2004.

5 T. Cayirli and E. Veral. Outpatient scheduling in healthcare: a review of literature. *Production and Operations Management*, 12:519–549, 2003.

6 Brian Denton and Diwakar Gupta. A sequential bounding approach for optimal appointment scheduling. *IIE Transactions*, 35:1003–1016, 2003.

7 Satoru Fujishige. *Submodular Functions and Optimization*, volume 58 of *Annals of Discrete Mathematics*. Elsevier, 2005. 2nd edition.

8 Linda V. Green, Sergei Savin, and Ben Wang. Managing patient service in a diagnostic medical facility. *Operations Research*, 54:11–25, 2006.

9 Diwakar Gupta. Surgical suites' operations management. *Productions and Operations Management*, 16:689–700, 2007.

10 Wiebke Höhn and Tobias Jacobs. On the performance of Smith's rule in single-machine scheduling with nonlinear cost. In *LATIN*, pages 482–493, 2012.

11 Oscar H. Ibarra and Chul E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of the ACM*, 22:463–468, 1975.

12 Satoru Iwata. Submodular function minimization. *Mathematical Programming*, 112:45–64, 2008.

13 Guido C. Kandoorp and Ger Koole. Optimal outpatient appointment scheduling. *Health Care Management Science*, 10:217–229, 2007.

14 Nicole Megow and José Verschae. Dual techniques for scheduling on a machine with varying speed. In *ICALP*, pages 745–756, 2013.

15 James B. Orlin. A faster strongly polynomial time algorithm for submodular function minimization. *Mathematical Programming*, 118:237–251, 2009.

16 Maurice Queyranne. Structure of a simple scheduling polyhedron. *Mathematical Programming*, 58:263–285, 1993.

17 Lawrence W. Robinson and Rachel R. Chen. Scheduling doctor's appointments: Optimal and empirically-based heuristic policies. *IIE Transactions*, 35:295–307, 2003.

18 W. E. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3:59–66, 1956.

19 Sebastian Stiller and Andreas Wiese. Increasing speed scheduling and flow scheduling. In *ISAAC*, pages 279–290, 2010.

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

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