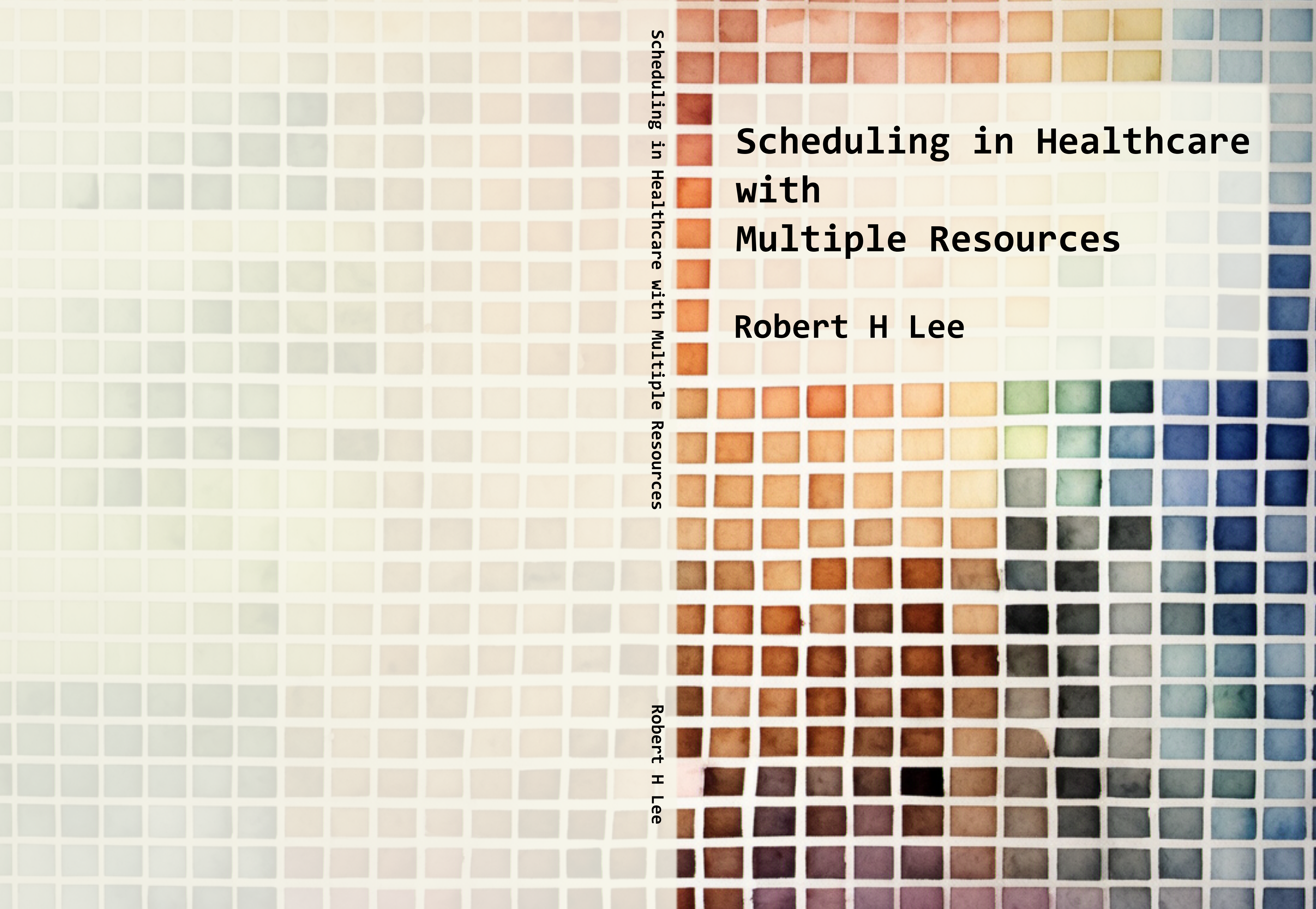
Scheduling in healthcare with multiple resources

Lee, R.H.

Publication date
2024

Document Version
Final published version

Citation for published version (APA):
Lee, R. H. (2024). *Scheduling in healthcare with multiple resources*. [Thesis, fully internal, Universiteit van Amsterdam].



Scheduling in Healthcare with Multiple Resources

Robert H Lee

Scheduling in Healthcare with Multiple Resources

Robert H Lee

Scheduling in Healthcare with Multiple Resources

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties ingestelde commissie,
in het openbaar te verdedigen in de Aula der Universiteit
op woensdag 8 mei 2024, te 14.00 uur

door Robert Henry Lee
geboren te Londen

Promotiecommissie

Promotores:	prof. dr. ir. D. den Hertog	Universiteit van Amsterdam
	prof. dr. R.J.M.M. Does	Universiteit van Amsterdam
Copromotores:	dr. A. Kuiper	Universiteit van Amsterdam
Overige leden:	prof. dr. M.R.H. Mandjes	Leiden University
	prof. dr. J. de Mast	Universiteit van Amsterdam
	prof. dr. J.A.S. Gromicho	Universiteit van Amsterdam
	prof. dr. S.I. Birbil	Universiteit van Amsterdam
	dr. ir. A. Braaksma	University of Twente
	dr. C. Zacharias	University of Miami

Faculteit Economie en Bedrijfskunde

Scheduling in Healthcare with Multiple Resources

Robert H Lee

Paranymphs:
Ujjwal Sharma and Paulina von Stackelberg

University of Amsterdam
8 May 2024

עֲשֵׂה לְךָ רַב וּקְנֵה לְךָ חָבֵר

Contents

Chapter 1

Introduction

1.1 Motivation

The need for optimization in healthcare is driven in no small part by the rapidly rising costs of healthcare. In more economically developed countries, these rising costs are met in part by increased spending on healthcare as a percentage of gross domestic product (GDP). Take, for example, the Netherlands, where this rose from 7.7% in 2000 to just over 11% in 2020 (World Health Organization 2023). In less economically developed countries, rising healthcare costs are particularly concerning as spending on healthcare as a percentage of GDP often does not increase accordingly, and in the case of India has even decreased. In the South Asian countries of Bangladesh, India and Pakistan, together home to more than 1.8 billion people, this figure was below 3% as of 2020 (World Health Organization 2023). As a result, in these countries, budgets remain the same while costs continue to increase, necessitating even the rationing out of healthcare services (Fazal et al. 2022).

Healthcare provided to individuals can be divided into the levels *primary* and *secondary care*. With some local modifications, the following is the system applied in, among others, the Netherlands (Schäfer et al. 2010), the United Kingdom (NHS 2022) and the United States of America (Phillips 2005). The first port of call for a patient is primary care, often known as a general practitioner or family doctor. If the patient's complaint cannot be resolved at this level, a referral follows to secondary care, where the patient will first be seen at an outpatient department. After consultation at an outpatient department, a decision is made to discharge the patient, book a follow-up appointment, or refer the patient to surgical care or other specialist care. Surgical care can be either outpatient surgery, in which case the patient is discharged the same day, or inpatient surgery, in which case the patient remains in a ward overnight for observation. While surgery is counted as part of secondary care, it will be convenient for the purposes of this dissertation to distinguish between *primary care*, *outpatient care*, and *surgical care*, due to differences in optimization approaches between these settings. We consider the recovery ward to be in support of surgery, so for the purposes of this dissertation group it under surgical care. Specialist inpatient services other than a recovery ward can be, for example, intensive care or neonatal care. Note that we do not consider either such inpatient services or emergency care in this dissertation, as these are special settings with their own unique demands.

At the primary care and outpatient care levels we are mostly concerned in this dissertation with the effective distribution of a set amount of time and capacity; to be more specific, how frequently and in what order patients should be seen by a doctor so as to minimize the healthcare provider's wasted time, while avoiding excessive waiting times for patients. In surgical care, not only are healthcare provider's wasted time and the patient's waiting time important, but the incredibly high costs of running a surgical suite and caring for patients before and after surgery must also be addressed.

Alongside this provision of healthcare to individuals, we also find large scale public health initiatives, the most extreme example of which no doubt being the complete eradication of smallpox in the wild via a worldwide vaccination program. Of course, such programs are incredibly expensive to run and are concerned with reaching as many people as possible, while not overwhelming the facilities available.

Having motivated the need to improve efficiency in healthcare and having established the scope of this dissertation, we will now take a step back from just primary, outpatient, and surgical care to look at a wider range of healthcare settings to which optimization has been applied. These settings include, among many other examples: the placement of blood distribution centers; the testing of donated blood for disease; the choosing of nurse-staffing levels; the locating (and relocating) of ambulances; the scale of healthcare provision for clinics; the allocation of surgeries to operating rooms; and the creation of appointment schedules. (respectively, Wemelsfelder et al. 2022, Bar-Lev et al. 2017, Kortbeek et al. 2015, Van Buuren et al. 2018, Zacharias and Armony 2017, Denton et al. 2010a, Ahmadi-Javid et al. 2017).

This call to improve efficiency comes with a cautionary tale. Decision makers should carefully consider what it is that they want to minimize or maximize, and ask whether this choice achieves their wider goals and what its unintended consequences may be. For the past twenty years or so, improving efficiency in healthcare in the Netherlands has meant minimizing costs. This has come at the expense of capacity and flexibility, which were much needed when the Dutch healthcare system was hit by a severe demand shock in the spring of 2020 (Kuiper et al. 2022).

The techniques laid out in this dissertation aim to contribute to the existent literature and applications for improving efficiency in healthcare, enabling limited funds and time to be used more effectively. For primary care, outpatient care, and large scale public health initiatives, this will be done by providing appointment schedules which make optimal use of limited resources such as the time available or the number of healthcare providers. And, at the level of surgical care, it will be done by showing how a master surgery schedule can be designed and implemented which maximizes production and tackles the high costs of healthcare while conforming to a hospital's many constraints. We will now take a brief look at the basic details of appointment scheduling and master surgery scheduling. We then finish this introduction with an overview of how this dissertation contributes to the literature on healthcare efficiency.

1.2 The Appointment Scheduling Problem

The study of appointment scheduling begins with Welch and Bailey (1952) who, at a time when physician was king, pled in *The Lancet* on behalf of the patient:

> *"It is not uncommon to find that patients are there for over an hour before being seen by the doctor with whom they have an appointment. During much of that time many just sit, often under conditions which do not permit the time being usefully or even pleasantly occupied. To keep patients waiting longer than is really necessary is clearly undesirable on humanitarian grounds."*

Of course, Welch and Bailey were not indifferent to the concerns of the doctor:

> *"We propose ... to show how an appointment system can be used to save the time of the patient without wasting the time of the consultant."*

That is, Welch and Bailey wanted to show that with the clever determination of when patients should arrive, one could drastically reduce the *waiting time* of patients while hardly increasing the *idle time* of the physician, this being time the physician spends waiting for a patient to arrive.

We will now take a look at one of the most simple formulations of the appointment scheduling problem, and the problem upon which three out of the four following chapters in this dissertation expand. Recall that we wish to find a balance between a patient's waiting time and a physician's idle time. Let there be n patients, numbered $i = 1, ..., n$. Let $\boldsymbol{x} = (x_1, x_2, \dots, x_n)$ be a vector of non-negative interarrival times, i.e. the time between when a patient i and a patient $i+1$ arrives. It is a convention to begin the day with the first patient already in place, which we do by letting $x_1 = 0$. Let the *random variables* B_i be the service time associated with patient i (the time the patient spends with the doctor, also called busy time), W_i be patient i's waiting time, I_i be the period of idle time experienced just prior to patient i's arrival, and S_i be the sojourn time of patient i (the total time that the patient spends in the system, that is both in waiting and in service). Lastly, let $0 < \omega \leq 1$ be the importance attributed to idle time. A starting point for the appointment scheduling problem is then to minimize a combination of expected idle and waiting times as given by the expression:

$$\min_{\boldsymbol{x}} \sum_{i=1}^{n} \omega \mathbb{E}[I_i] + (1-\omega)\mathbb{E}[W_i], \tag{1.1}$$

where $\mathbb{E}[I_i]$ and $\mathbb{E}[W_i]$ can be calculated from the random variables found via the Lindley recursion (Lindley 1952):

$$
\begin{aligned}
S_i &= B_i + W_i, \\
I_{i+1} &= \max\{x_i - S_i, 0\}, \\
W_{i+1} &= \max\{S_i - x_i, 0\}.
\end{aligned}
$$

As already mentioned, by convention we suppose that we start the day with the first patient already in place, $x_1 = 0$, and as this patient will see no waiting time, nor incur any idle time, we also let $I_1 = W_1 = 0$.

An optimal solution to this problem, $\boldsymbol{x}^*$, has a so-called *dome-shaped* pattern, where the lengths of the interarrivals increase at the beginning of the schedule, tending towards a plateau, after which they gradually decrease (Ahmadi-Javid et al. 2017).

Note that this is a multidimensional optimisation problem with n customers and $n-1$ decision variables, due to x_1 being fixed. Note also that the sojourn time $S_i = B_i + W_i$ is the convolution of two random variables, which in general cannot be expressed in closed form. The art is thus in finding an appropriate representation for sojourn times. To give some flavor, this might be done by assuming exponentially distributed (and thus memoryless) service times as in Jansson (1966), by phase types as in Kuiper and Lee (2022), heuristic methods such as in De Kok (1989) and Lee and Kuiper (2024), or, of course, by simulation.

A convenient feature of the Lindley recursion is that the objective function (1.1) is convex in $\boldsymbol{x}$ and so we are guaranteed to eventually find a minimum (Kuiper et al. 2023). This recursion, however, may only be used in the case of a single server, or in healthcare terms under the assumption of *continuity of care*: that one patient will always see the same practitioner. In this dissertation, this assumption is relaxed, and not only must we consider how to model sojourn times, but we are also confronted by the question of whether the problems we tackle are convex and thus whether we can guarantee that an optimum is found.

Another common assumption that we relax is that all patients have identically distributed service times to one another. If this is not the case, then the order in which patients $1, 2, \ldots, n$ are seen is also important. This is the matter of sequencing, which we will also address and which has been shown through experimentation to in fact have more impact on the quality of the solution than the schedule itself (Çayırlı et al. 2006).

Above, we listed some approaches for dealing with the randomness in appointment scheduling. These methods all require an assumed probability distribution for patients' service times. If this distribution in reality deviates from the assumption, then the quality of the solution could suffer. This motivates the use of *robust optimization* to minimize the worst-case performance of an appointment schedule. The most frequently applied robust method in appointment scheduling literature is *distributionally robust optimization.* Under this method, one aims to minimize the expected cost as in display (1.1) subject to a *worst-case distribution* based usually upon the first two moments of the service time distribution. This method has found application in works by Mak et al. (2015) Zhang et al. (2017), and Lee and Kuiper (2024).

Robust optimization also offers an alternative to appointment scheduling in the form of *robust appointment scheduling*; a method which replaces the stochastic elements entirely with a so-called *uncertainty set*, defined, for example, by lower and upper bounds on how long an appointment might last. One then finds a solution which has the lowest worst-case performance across all possible realisations within the uncertainty set. This approach has been developed and utilized by Mittal et al. (2014), Schulz and Udwani (2019), Bandi and Gupta (2020), and Gao et al. (2022).

1.3 The Master Surgery Scheduling Problem

In surgical care, not only is time limited, but we must also consider operating rooms, surgeons, other operating room personnel, recovery ward personnel, and limited and expensive equipment. Creating a schedule which takes all these resources into account quickly becomes very complex. To keep track of these assignments, a hospital will make use of a master surgery schedule, a tool which assigns surgical specialties (e.g., urology) or even individual surgeons to a particular operating room at a particular time. This combination of room and time is called a *block*, such that specialties or surgeons are assigned to blocks.

The choice of how many and which types of operations to perform within a block is generally left open, and specialties often make these decisions for themselves. The leaving open of this decision places master surgery scheduling at the *tactical* level of decision making. We can distinguish three such levels of decision making: *Strategic* decisions, such as how many operating rooms to maintain, constrain *tactical* decisions, such as which specialty should be assigned an operating room, which in turn constrain *operational* decisions, such as which surgery is to be performed when. This last level is also the level at which appointment scheduling resides, as its solutions dictate schedules on a day-by-day or even minute-by-minute basis. It should be reiterated that appointment scheduling is employed in primary and outpatient care; the related field of *surgery scheduling* (Guda et al. 2016), which we touch upon only briefly in this dissertation, is its analogue within surgical care.

There are three main strategies to assign specialties to blocks, all forms of master surgery schedule (Fei et al. 2010). *Open scheduling* is a first come first served system (Main 1995) where all blocks are "released" several months in advance and any surgeon may claim any open block in which to operate. *Block scheduling* is a strategy where specialties are assigned to specific blocks usually within a cyclic schedule, that is one which repeats, with a four-week cycle being a common choice. This cyclic schedule may remain in place for multiple years, with little if any modification during that time. The last strategy is *modified block scheduling*, which is built upon block scheduling, but which allows a fraction of blocks to be treated as if in a first come first served system. Each of these of course has its advantages and disadvantages, although block and modified block scheduling continue to be the most frequently applied (Fei et al. 2010). Unfortunately, there is a shortage of applied works in literature, so it is difficult to ascertain why particular scheduling strategies are chosen, what decisions hospitals frequently face, or the reasons why the development of a given schedule succeeded or failed (Cardoen et al. 2010).

In this dissertation we describe in detail the development and implementation of a master surgery schedule for a medium sized regional hospital in the Netherlands. This schedule makes use of a block scheduling strategy, and we describe the development and implementation of this schedule in detail.

1.4 Overview of the Contribution of this Dissertation

In Chapter 2 we consider the joint question of sequencing and scheduling, that is, we ask in what order patients with differing characteristics in service time should arrive, and how much time there should be between two arrivals. We consider two variants of the scheduling problem in a setting with one physician: that from display (1.1) and the *sequential problem* of Kemper et al. (2014). Applying the heuristic of De Kok (1989) to calculate the convolutions required for the sojourn times S_i, we derive sequencing rules to be used in both cases. We also show an equivalence between objectives of the sequential problem and the *surgery scheduling problem* (Guda et al. 2016), which concerns the scheduling of individual patients to surgery, and is distinct from the master surgery scheduling problem. Finally in this chapter, we present robust optimization results for the sequential problem, namely we show how to minimize the worst-case expected waiting and idle times both when the distribution of service times is not known, and when the ideal weight to place on idle time is not known.

This work is relevant for the primary and outpatient care settings where both *new* and *return* patients are often scheduled, with new patients in general having both greater expected service times and variance in service time. This approach can also be extended to planning the order and duration of surgeries thanks to the increased use of statistical learning techniques for estimating surgery duration.

In Chapter 3 we investigate the effect of relaxing *continuity of care* within a healthcare setting by considering, by means of phase-type distributions, schedules in which a patient may be seen by any one of multiple healthcare providers. In this chapter we report significant savings that can be had from pooling healthcare, and exhibit the shapes of the optimal schedules, which are no longer guaranteed to always follow the dome-shaped pattern mentioned earlier.

The impressive saving that can be had from pooling healthcare providers motivates the claim that unless continuity of care is strictly necessary in a given setting, it should be abandoned. This chapter presents a general method for doing so that is suitable for all reasonable means and variances of service time distribution that occur in appointment scheduling.

Chapter 4 takes the theme of Chapter 2 one step further by considering the case where one not only has multiple providers, but also permits multiple patients to arrive at one time. We examine this setting through queueing theoretical approaches, namely assuming exponential service times and by looking at the system in steady state. For this setting, we are able to prove convexity of the objective function, a combination of expected idle and waiting times. This setting enjoys nearly the same savings as in Chapter 2's setting, while having certain advantages in ease of implementation versus Chapter 2's setting, such as a unique appointment book per physician.

This chapter is inspired by large scale settings such as testing and vaccination programs in a public health context, but also extends to more general settings, such as timed entry slots to museums, sports facilities, or even exam reviews.

Chapter 5 differs from the previous three chapters in that while they focus on appointment scheduling, Chapter 5 details the development and implementation of a master surgery schedule for a medium sized hospital in the Netherlands. The hospital's primary objective was to minimize the number of *split blocks*. A split block occurs when a surgical specialty is scheduled to either the morning or the afternoon, but not the full day. Split blocks are undesirable, as they require refitting the operating room between specialties and thus eat into productivity. When assigned a whole block, a specialty can operate throughout the entirety of the working day.

Mixed integer linear optimization was used to tackle this problem and provided a solution which reduced the number of split blocks from 40 to 14 across a 4 week cyclical schedule. This improvement frees up surgery time for 50 patients a year, or 300 000 Euro in revenue. This project demonstrates both the pressure felt by healthcare providers to become more efficient, and a method that aids in achieving this goal.

The contribution of this chapter is threefold: it provides valorization through the marked improvement of the master surgery schedule at the Red Cross Hospital; it provides an exposé on a successful application of Operations Research in practice, which as mentioned by Cardoen et al. (2010) is particularly lacking in the field of master surgery scheduling; and, lastly, it provides a theoretical contribution in demonstrating a method of tackling the symmetries inherent in many linear programming models of the master surgery scheduling problem.

1.5 Personal Contribution to Chapters

This dissertation is based upon the four following pieces:

Chapter 2 Lee, R. H. and Kuiper, A. (2024). Optimal sequencing using a scheduling heuristic. *Computers & Operations Research*, 161.

Chapter 3 Kuiper, A. and Lee, R. H. (2022). Appointment scheduling for multiple servers. *Management Science*, 68(10):7422–7440.

Chapter 4 Lee, R.H. and Kuiper, A. On the design of appointment books in the case of multiple servers. *Work in process.*

Chapter 5 van Ham, V., Lee, R. H., and Kuiper, A. (2023). Optimizing and implementing a new master surgery schedule: Increasing productivity and balancing outflow. *Available at SSRN: abstract id 4634128.*

Each chapter contains ideas from all of its contributors. All analysis and numerical experiments of Chapters 2, 4, and 5 are my own, with those of Chapter 3 being joint work. The writing of Chapters 2 and 4 is my own, while the writing of Chapters 3 and 5 is joint.

Chapter 2

On Sequencing and Scheduling

2.1 Introduction

Since the introduction of the scheduling problem in Welch and Bailey (1952), the literature has predominantly focused on finding schedules that minimize a linear combination of the sum of expected idle and waiting times. Idle time is defined as the time that a server (e.g., a physician) has to wait for clients, and waiting time as the time that a client has to wait before being served. The difficulty lies in the fact that service-time durations are random, for example, following a log-normal distribution (May et al. 2000, Çayırlı et al. 2006). This makes the problem generally intractable, although it can be expressed in brief as

$$\begin{aligned}\bar{\boldsymbol{x}} &= \arg\min_{\boldsymbol{x}} \mathscr{F}(\boldsymbol{x};\omega) \\ &= \arg\min_{\boldsymbol{x}} \omega \sum_{i=1}^{n} \mathbb{E}[I_i(\boldsymbol{x})] + (1-\omega) \sum_{i=1}^{n} \mathbb{E}[W_i(\boldsymbol{x})], \end{aligned} \tag{2.1}$$

in which $\omega \in (0,1)$ is a value chosen by the practitioner, and I_i and W_i are idle and waiting times associated with the ith client. The objective function $\mathscr{F}(\boldsymbol{x};\omega)$ is minimized over the possible *inter-arrival times* $\boldsymbol{x} = (x_1,\dots,x_{n-1})$, which determine scheduled arrival epochs by $t_j = \sum_{i=1}^{j-1} x_i$ for $j = 1,\dots,n$ with the convention that an empty sum equals zero, so that $t_1 = 0$.

The scheduling problem displayed in Equation (2.1) is computationally expensive, requiring multi-dimensional optimization and a series of convolutions to determine clients' sojourn-time distributions. Alternatively, one can use the sequential approach of Kemper et al. (2014) which permits an analytical solution and can be solved without the use of optimization packages.

$$x_i^* = \arg\min_{x_i} \mathscr{F}_i(x_i;\omega) = \arg\min_{x_i} \mathbb{E}[\omega\, I_{i+1}(x_i) + (1-\omega)\, W_{i+1}(x_i)]. \tag{2.2}$$

Analogously, for surgery scheduling the problem is framed as minimizing an objective in terms of earliness E_j and tardiness T_j, which are related to the client's completion times C_j and due dates d_j, e.g., Guda et al. (2016). The optimal due dates $\bar{\boldsymbol{d}}$ are to be found as a result of, again, a minimization:

$$\bar{\boldsymbol{d}} = \arg\min_{\boldsymbol{d}} \mathscr{G}(\boldsymbol{d};\omega) = \arg\min_{\boldsymbol{d}} \sum_{j=1}^{n} \mathbb{E}\left[\omega\, E_j(d_j) + (1-\omega)\, T_j(d_j)\right]. \tag{2.3}$$

The key difference between the surgery scheduling problem and that of appointment scheduling is that the former assumes no delays or server idling: once a surgery is completed the next one starts right away (Guda et al. 2016). However, we show that the problems displayed in equations (2.1) and (2.3) are equivalent when one composes a schedule sequentially, i.e., the setting displayed in (2.2).

The application of a moment-iteration method which uses the first two moments of a client's service time distribution to iteratively build each client's sojourn-time distribution is extremely scalable. This approach also permits clients with heterogeneous service time distributions, in which case one must also consider the sequence in which clients are scheduled. Finding the best sequence to minimize cost is deemed one of the most important open problems in the field of scheduling, see Ahmadi-Javid et al. (2017). Using the moment-iteration method we derive optimal sequencing rules, which augment results presented in Wang (1999), Kemper et al. (2014) and Choi and Wilhelm (2020); showing under sequential optimization and with exponential service times that lowest mean (or variance) first is optimal and classifying in which cases similar rules hold for the log-normal distribution.

In the next section, we review relevant literature on the typical appointment scheduling problem. In the subsequent section, Section 2.3, we compare simultaneous and sequential schedules, highlighting various merits of sequential scheduling, such as the resemblance to the problem of due date determination as common in surgery scheduling. Thereafter, in Section 2.4, we elaborate on the moment-iteration method applied to our approach, specifically to the case of exponentially and log-normally distributed service times, allowing tractability. In Section 2.5, we show how to combine sequential schedules with the moment-iteration method. In Section 2.6, we examine how varying the moments (mean and standard deviation) of the service-time distribution affects the optimal cost for sequential schedules, allowing the generation of sequencing rules. In Section 2.7, we assess the approach numerically, and, in Section 2.8, we extend the approach in several directions including the extension to the case when no distributional information is available. We conclude and discuss our results in Section 2.9.

2.2 Literature Review

For comprehensive reviews on appointment scheduling and optimization methods developed we refer to Çayırlı and Veral (2003) and Ahmadi-Javid et al. (2017). Below we point out works that are especially relevant to our contribution.

2.2.1 Modelling Service Times

In this paper we primarily focus on the case of exponential or log-normally distributed service times. The former has attractive properties, i.e., memorylessness, and is often used in literature, see for example Kaandorp and Koole (2007) and Hassin and Mendel (2008). The latter, however, is the common choice when modelling real-life service times; for which we further refer to Klassen and Rohleder (1996) for an exposition on theoretical grounds and empirical evidence. Also, in recent papers, the log-normal distribution is used to model service-time distributions of both *new* and *return* clients (Çayırlı et al. 2006, Table 2) and Çayırlı et al. (2008, Section 2.3). Similarly, surgery durations are often modelled by means of log-normal variables, see May et al. (2000).

We will introduce a method of iteratively re-estimating the waiting-time distribution by matching moments to the field of optimized appointment scheduling. This method approximates intractable convolutions for waiting times, which can only be numerically evaluated by reducing the *lag* to which extent you take the history into consideration (Vink et al. 2015). De Kok (1989) shows that this *moment-iteration method* results in accurate approximations for $G/G/1$ queues for the typical distributions chosen in queueing theory. Adan et al. (1995) use this method to examine $D/G/1$ queues with discrete service time distributions finding 'excellent' performance. As our method relies on the procedure outlined by De Kok (1989), we also refer to the work of Fenton (1960), who shows that a sum of log-normally distributed variables can be approximated well by a single log-normally distributed variable by matching the first two moments. Furthermore, based on their experiments, Ho and Lau (1992) state that performance of appointment scheduling rules is unaffected by the skewness and kurtosis of the service-time distribution. Indeed, Kuiper et al. (2015) achieve good performance when matching phase-type distributions to approximate service-times based on the first two moments only.

2.2.2 Sequential Scheduling

The solution from Eq. (2.1) must be found with numerical methods. As each element x_i of the vector $\boldsymbol{x}$ is solved for concurrently, we refer to such a solution as a *simultaneous* schedule. The resulting schedules have a notable dome shape, with inter-arrival times that first increase and then decrease towards the end of the session (Stein and Côté 1994, Wang 1999, Denton and Gupta 2003, Hassin and Mendel 2008, Kaandorp and Koole 2007, Kuiper et al. 2015).

Instead of jointly minimizing the objective function in Eq. (2.1), optimization can also be performed in a sequential manner, as studied in Wang (1993), Kemper et al. (2014), and Kuiper et al. (2015). In this approach the ith inter-arrival time x_i is determined to minimize a weighted sum of idle and waiting times of only its successor. In this way the problem can be recognized as iteratively solving a *newsvendor* problem (see also Weiss 1990, Mak et al.

2014):

$$\mathscr{F}_i(x_i;\omega) = \omega\,\mathbb{E}[I_{i+1}(x_i)] + (1-\omega)\,\mathbb{E}[W_{i+1}(x_i)]$$

for a specific choice of $\omega \in (0,1)$, with waiting times and idle times to be seen as *overage* and *underage*. So, for $i = 1, \dots, n-1$ the following partial objective function is solved *iteratively*

$$x_i^* = \arg\min_{x_i} \mathscr{F}_i(x_i;\omega) = \arg\min_{x_i} \omega\,\mathbb{E}[I_{i+1}(x_i)] + (1-\omega)\,\mathbb{E}[W_{i+1}(x_i)].$$

The result is a *sequentially* optimized schedule in which expected idle and waiting times grow proportionately throughout the schedule. This lends a uniformity of performance measures which, as is pointed out in Çayırlı and Veral (2003), may be desirable, especially when compared to the performance measures resulting from simultaneous optimization. Similarly, Ho and Lau (1992) find that schedules with fixed intervals are unfair in the sense that clients scheduled earlier experience less waiting time. They thus propose to vary appointment intervals such that clients *early* in the session are to arrive *earlier*, whereas clients *later* in the session should arrive *later*, which is accounted for in the scheduling rule proposed by Yang et al. (1998), reducing the variance of the waiting times.

One can also consider some alternative objectives that might achieve a similar goal. For example, Millhiser and Veral (2015) argue in favor of a policy which restricts the probability of a client experiencing an excessive waiting time, i.e., a waiting time of greater than 20 min, which corresponds to using a quantile objective as studied in Sang et al. (2021). Yan et al. (2015) impose service fairness on the schedule by implementing a constraint formulated as the difference between the maximum and minimum average waiting time among clients.

2.2.3 Sequencing Clients

Sequencing of clients is recognized as one of the key open challenges in appointment scheduling (Ahmadi-Javid et al. 2017); the importance for practice is emphasized in Vanden Bosch and Dietz (2000) who compare the objective function under pair-wise swaps and distil optimal policies for dealing with heterogeneous client groups. Furthermore, Çayırlı et al. (2006, 2008) show in their experiments that sequencing has a more profound impact on performance than deriving the optimal schedule. Salzarulo et al. (2011) show for a large primary care facility that the performance of an appointment schedule can be improved from 5% up to 25% through the application of sequencing rules.

Various papers have claimed that sequencing in order of Smallest Variance First (`SVF`) is optimal, e.g., Klassen and Rohleder (1996), Denton et al. (2007), and Mak et al. (2015) for robust schedules. However even for two clients, which is equivalent to the classical newsvendor problem, it is shown that counterexamples can be constructed, see Ridder et al. (1998). Recently, De Kemp et al. (2021) provide worst-case bounds on the performance of `SVF`.

Mancilla and Storer (2012) use sample-average approximations in combination with linear programming to show that their solutions result in a (small) improvement over the SVF heuristic in a practical setting. Also, Kong et al. (2016) show by constructing counterexamples, often with a considerable number of clients, that the SVF rule is not optimal. Since most of their examples follow an equidistant schedule for tractability, they call for more research when schedules are optimized over the arrival times as well. Jafarnia-Jahromi and Jain (2020) show through the use of counterexamples and by defining equivalence classes that a general optimal sequencing rule for the appointment scheduling problem does not exist.

Besides introducing the moment-iteration method to the field of appointment scheduling, this paper derives, under the sequential optimization paradigm, rigorous sequencing rules which go further then simplistic settings of only 2 or 3 clients, see Weiss (1990) and Gupta (2007). It specifically extends on the exponential case as studied in Wang (1999) and Choi and Wilhelm (2020) who, also considering the exponential case, show that SVF is best when schedules are composed as cumulative sums of expected service times of prior clients. These schedules are called *proportional* and bear a resemblance to the sequential optimization framework; the authors even call for extending their work to the log-normal case. Furthermore, Kemper et al. (2014) study sequencing, but focus only on scale families which have the restriction that the coefficient of variation equals a constant. Our approach and sequencing rules carry over to surgery scheduling, as there is a direct connection between the problem of appointment scheduling and that of surgery scheduling, in which earliness and tardiness are minimized by finding optimal due dates. In that regard we also augment the results of Guda et al. (2016), who show for various distributions that the SVF heuristic is optimal.

2.3 Sequential Optimization Modelling Framework

The challenge in appointment scheduling is to determine arrival times for n clients, for example, by minimizing an objective function as displayed in Eq. (2.1). Another approach that aligns well with the underlying idea of the moment-iteration method is sequential optimization, which *iteratively* solves the partial objective, permitting an analytical solution and which can thus be programmed without the use of optimization packages. Furthermore, it has some attractive features (full control of cost, a natural online version, and direct connection to surgery scheduling) and combined with the moment-iteration method it results in a fast and scalable optimization procedure, for which sequencing rules can be derived.

2.3.1 Determining Inter-Arrival Times in Appointment Scheduling

To fully understand the benefits of the sequential optimization approach of Eq. (2.2), note that it can be rewritten as:

$$x_i^* = \arg\min_{x_i} \omega \int_0^{x_i} F_{S_i}(s)\ ds + (1-\omega) \int_{x_i}^{\infty} (1 - F_{S_i}(s))\ ds, \tag{2.4}$$

where $F_{S_i}(\cdot)$ denotes the cumulative distribution function of S_i. Using the above equation we observe that the per-client optimization procedure depends fully on the sojourn-time distribution of his or her predecessor. In that sense such a schedule is a sequential optimization problem and it holds that all $n-1$ optimization problems are *convex* in x_i. The optimal inter-arrival times for the problem are now found by taking the derivative in Eq. (2.4) and setting it equal to zero to arrive at

$$x_i^* = F_{S_i}^{-1}(1-\omega). \tag{2.5}$$

A computational advantage of sequential optimization is that it does not require the use of opaque numerical methods which can confound interpretation, and is much faster to calculate. Finally, by combining sequential optimization with a moment-iteration method, which is described in the next section (Section 2.4), both a further speed-up of calculation and optimal sequencing rules can be attained. Furthermore, the approach allows a new client to be appended to the current schedule if required without impairing optimality—and thus can also be used *online*. Finally, the sequentially optimized inter-arrival times coincide with departure times in surgery scheduling, allowing many of our results to carry over to the problem of surgery scheduling, as shown in the next section, Section 2.3.2.

2.3.2 Equivalence with Due Date Determination in Surgery Scheduling

Soroush (1999) introduced the problem of the single-machine earliness/ tardiness problem with stochastic job durations (SET). The goal in such a setting is to find the optimal due dates that minimize a linear combination of early and late times. Guda et al. (2016) applied this framework to operating rooms. They argue that, due to tremendous scarcity of these facilities, idle time is entirely eliminated and as a consequence all jobs (clients) are available from the start of the session. As a consequence, each surgery begins as soon as the previous one is finished, resulting in a problem in which only due dates which minimize *earliness* and *tardiness* need to be determined. Idle times are fully neglected in their approach, but they do call for a unifying method which includes idle and waiting times. Interestingly enough the two problems, surgery scheduling and appointment scheduling, are equivalent under the sequential optimization approach, as we will show.

Let $C_i = \sum_{j=1}^{i} B_j$ be the completion time of client i, and d_i be the planned due date of client i. So, earliness of client i as $E_i = (d_i - C_i)^+$ and tardiness as $T_i = (C_i - d_i)^+$ are the inputs of the objective in Eq. (2.3). Letting F_{C_i} be the CDF for the completion time of client i, the optimal due date is $d_i^* = F_{C_i}^{-1}(1-\omega)$, analogous to the sequential optimization case, cf. Eq. (2.5). However, in the surgery scheduling setting, there is no idling from subsequent clients having yet to arrive.

Nevertheless, there is a link between these problems. In a sequentially optimized schedule the optimal arrival time of client i, chosen to minimize a linear combination of idle and waiting time, is equal to the due date of client $i-1$, chosen to minimize a linear combination of earliness and tardiness with the same coefficients. We present and prove the statement formally in the next proposition.

Proposition 2.1. *Under the sequential scheduling paradigm the objective and optimal solution of the appointment scheduling problem and surgery scheduling are equivalent, i.e.,*

$$\min_{x_i} \mathscr{F}_i(x_i;\omega) = \min_{d_i} \mathscr{G}_i(d_i;\omega),$$

so that:

$$x_i^* = \arg\min_{x_i} \mathscr{F}_i(x_i;\omega) = \arg\min_{d_i} \mathscr{G}_i(d_i;\omega) = d_i^*.$$

Proof. Writing out the partial objective $\mathscr{G}_i$ of the surgery scheduling problem from Eq. (2.2) in terms of earliness and tardiness gives:

$$\begin{aligned}\min_{d_i} \mathscr{G}_i(d_i;\omega) &= \min_{d_i}\ \mathbb{E}\left[\omega\, E_i + (1-\omega)\, T_i\right] \\ &= \min_{d_i} \mathbb{E}\left[\omega\,(d_i - C_i)^+ + (1-\omega)\,(C_i - d_i)^+\right].\end{aligned}$$

In appointment scheduling the completion time of client i is its arrival time plus its sojourn time: $t_i + S_i = \sum_{j=1}^{i-1} x_j + S_i$, so that the problem becomes

$$\begin{aligned}\min_{d_i} \mathscr{G}_i(d_i;\omega) = \min_{d_i}\ \mathbb{E}\ &\left[\omega\left(\left(d_i - \sum_{j=1}^{i-1} x_j\right) - S_i\right)^+\right. \\ &\left.+\ (1-\omega)\left(S_i - \left(d_i - \sum_{j=1}^{i-1} x_j\right)\right)^+\right].\end{aligned}$$

Define $x_i :- d_i - \sum_{j=1}^{i-1} x_j$ and recalling that under the sequential optimization framework the inter-arrival times of previous clients are already determined ($d_i = x_i + \sum_{j=1}^{i-1} x_j^*$), we have:

$$\min_{d_i} \mathscr{G}_i(d_i;\omega) = \min_{x_i}\ \mathbb{E}[\omega I_{i+1}(x_i) +\ (1-\omega) W_{i+1}(x_i)]$$

$$= \min_{x_i} \mathscr{F}_i(x_i; \omega), \tag{2.6}$$

which completes the equivalence. □

Besides the fact that the sequential approach controls the distribution of costs, the optimized inter-arrival times coincide with departure times. Eq. (2.6) shows that the objective is the same, but also the due date that is found for client i is the arrival epoch of the next client, i.e., the sum of the first i inter-arrival times. So, the point at which it is optimal for the new client to arrive matches the point at which it is optimal for the preceding client to depart. This implies that tardiness corresponds to waiting, and earliness to idling.

2.3.3 Including Overtime and No-Shows

Besides idle and waiting times, overtime is also considered an important metric for assessing the performance of a schedule. It is defined as the time that a session is prolonged beyond its scheduled end time. Recall that C_i is defined as the time when the ith client is finished, which is mathematically defined as:

$$C_i = \sum_{j=1}^{i} (B_j + I_j)\,, \tag{2.7}$$

i.e., the sum of service and idle times. Overtime is defined as the time that a session exceeds a targeted session-end time, T. Consequently, it relates to the finish time of the last client, or the session's makespan via

$$O = \max\{C_n - \mathrm{T}, 0\}\,.$$

Noting that overtime is built of idle times, we can relate the overtime to the weight placed on idle time — a behavior which has previously been pointed out by Kuiper et al. (2023). This notion makes it possible to incorporate overtime in our sequential framework by emphasizing individual idle times more. In the case of overtime we can write the following objective function:

$$\mathscr{F}(\boldsymbol{x}; \omega, \gamma) = \omega \sum_{i=1}^{n} \mathbb{E}[I_i(\boldsymbol{x})] + (1-\omega) \sum_{i=1}^{n} \mathbb{E}[W_i(\boldsymbol{x})] + \gamma\, \mathbb{E}[O], \tag{2.8}$$

where γ is often set to 1.5 the value of idle time, i.e., 1.5ω (Çayırlı and Yang 2014). This is also the value that we shall be using in this paper. Note that $\mathscr{F}(\boldsymbol{x}; \omega) = \mathscr{F}(\boldsymbol{x}; \omega, \gamma)$ if and only if $\gamma = 0$ or $\mathrm{T} = \infty$.

Due to the iterative procedure of sequential optimization, it is difficult to elicit a term for overtime in the inter-arrival time for client i, x_i, and indeed impossible if one wants to maintain the elegance of the solution. However, to resolve this consider the case where $\mathrm{T} = 0$, then O reduces to just a sum of idle times, which provides an upper bound on the ω value. Alternatively, one can let T tend to infinity so that overtime never occurs and the original objective function is obtained, providing a lower bound. So, in summary, choosing the weight of expected overtime to be $\gamma = \alpha\omega$, e.g., $\alpha = 1.5$, we have the interval $(\omega, {}^{\omega(1+\alpha)}/{}_{1+\alpha\omega})$ in which some ω^o lies such that overtime is accounted for in the objective function.

To incorporate no-shows, note that since a no-show is essentially a service-time of 0 time units, we incorporate this source of randomness by adapting the mean and variance on which the schedule is built accordingly. Given a client's no-show percentage, q_i, the mean and variance can be modified by $\mathbb{E}[\bar{B}_i] = (1-q_i)\mathbb{E}[B_i]$ and $\mathbb{V}\text{ar}\,[\bar{B}_i] = (1-q_i)\mathbb{V}\text{ar}\,[B_i] + q_i(1-q_i)\mathbb{E}[B_i]^2$. These updated quantities can then be substituted in the approach, more specifically by adapting Equations (2.10) and (2.11). The first two moments can be similarly adapted for walk-ins, see Kuiper et al. (2023).

2.4 Moment Iteration Method

In this section we lay out the moment-iteration method for approximating the sojourn-time distributions. Define the ith client's service duration as B_i, and sojourn time as $S_i = W_i + B_i$, where $S_1 = B_1$, then by virtue of the Lindley recursion the idle and waiting times in the objective functions of (2.1) and (2.2) can be determined recursively, for $i = 1, \dots, n-1$:

$$I_{i+1} = \max\{x_i - S_i, 0\}, \quad \text{and} \quad W_{i+1} = \max\{S_i - x_i, 0\}. \tag{2.9}$$

Note that I_1 and W_1 are equal to zero as we assume that the session starts with the arrival of the first client. The method is based on approximating the convolution $S_i = B_i + W_i$ by fitting a distribution of the same type as chosen for the B_i s. Naturally, the number of moments to fit equals the number of parameters that determine the probability distribution function. In case of the exponential distribution that is only the mean, but typically in appointment scheduling the use of two moments is warranted. The log-normal distribution has two parameters, so that both the mean and variance need to be matched:

$$\mathbb{E}[S_i] = \mathbb{E}[W_i] + \mathbb{E}[B_i]; \tag{2.10}$$

$$\mathbb{V}\text{ar}\,[S_i] = \mathbb{V}\text{ar}\,[W_i] + \mathbb{V}\text{ar}\,[B_i]. \tag{2.11}$$

If a service time B_i for all $i = 1, \dots, n$ follows a log-normal distribution, i.e., $B_i \sim \mathcal{LN}(\mu_i, \sigma_i^2)$ then the μ_i and σ_i represent the location and scale parameter for the corresponding normally distributed variables: $\log(B_i)$. The probability density function of B_i is given by

$$f_{B_i}(t) = \frac{1}{t\sigma_i\sqrt{2\pi}}\,\mathrm{e}^{-\frac{(\log t - \mu_i)^2}{2\sigma_i^2}}\,\mathbf{1}_{(0,\infty)}(t).$$

So, at time zero, we have that $S_1 = B_1$. For $i = 2$, we approximate the sojourn-time distribution by a log-normal distribution with the same mean and variance as $B_2 + W_2$. We then use this newly fitted distribution for S_2 to likewise infer S_3 and so on for all $i \geq 2$, evaluating the parameters as in Equations (2.14) and (2.15). In this way, we have

$$\begin{aligned} S_i &\approx \mathcal{LN}(\nu_i, \tau_i^2) \\ &= \mathcal{LN}\left(\log\left(\frac{\mathbb{E}[S_i]^2}{\sqrt{\mathbb{V}\mathrm{ar}[S_i] + \mathbb{E}[S_i]^2}}\right), \log\left(1 + \frac{\mathbb{V}\mathrm{ar}\,[S_i]}{\mathbb{E}[S_i]^2}\right)\right). \end{aligned} \tag{2.12}$$

The laborious part is computation of the first two moments of the waiting-time distribution. For that purpose we give the kth partial moment of a random variable X as $m_X(a,k) = \int_a^\infty x^k f_X(x)\ dx$. Since S_{i-1} follows a log-normal distribution with parameters $(\nu_{i-1}, \tau_{i-1}^2)$ the kth partial moment of client i's sojourn time is given by

$$\begin{aligned} m_{S_i}(x_i,k) = \int_{x_i}^\infty s^k f_{S_i}(s)\ ds &= \int_{\log(x_i)}^\infty \mathrm{e}^{kz}\, \mathrm{e}^{\nu_i + \tau_i z}\, \varphi(z)\ dz \\ &= \mathrm{e}^{k\,\nu_i + \frac{1}{2}(k\,\tau_i)^2}\, \Phi\left(\frac{\nu_i - \log(x_i)}{\tau_i} + k\,\tau_i\right). \end{aligned} \tag{2.13}$$

where Z denotes a standard normally distributed variable, with density $\varphi(z)$ and cumulative distribution function $\Phi(z)$, so that the moments of the approximating log-normal distribution become

$$\begin{aligned} \mathbb{E}[S_i] &= \mathbb{E}[W_i] + \mathbb{E}[B_i] \\ &= \left(m_{S_{i-1}}(x_{i-1},1) - x_{i-1} m_{S_{i-1}}(x_{i-1},0)\right) \\ &\quad + \left(\mathrm{e}^{\mu_i + \frac{1}{2}\sigma_i^2}\right); \end{aligned} \tag{2.14}$$

$$\begin{aligned} \mathbb{V}\mathrm{ar}\,[S_i] &= \mathbb{V}\mathrm{ar}\,[W_i] + \mathbb{V}\mathrm{ar}\,[B_i] \\ &= \Big(m_{S_{i-1}}(x_{i-1},2) - 2x_{i-1} m_{S_{i-1}}(x_{i-1},1) + x_{i-1}^2 m_{S_{i-1}}(x_{i-1},0) \\ &\quad - \left(m_{S_{i-1}}(x_{i-1},1) - x_{i-1} m_{S_{i-1}}(x_{i-1},0)\right)^2\Big) \\ &\quad + \left(\mathrm{e}^{2\mu_i + \sigma_i^2}\left(\mathrm{e}^{\sigma_i^2} - 1\right)\right). \end{aligned} \tag{2.15}$$

For the purposes of optimization, we also wish to evaluate expected idle time $\mathbb{E}[I_i]$ and expected overtime $\mathbb{E}[O]$. The expected idle time for client i is straightforwardly calculated as

$$\mathbb{E}[I_i] = \mathbb{E}[S_i] - \mathbb{E}[B_i] + \sum_{j=1}^{i-1} \left(x_j - \mathbb{E}[B_j] - \mathbb{E}[I_j]\right), \tag{2.16}$$

where one can obviously maintain the term $\sum_{j=1}^{i-1}(x_j - \mathbb{E}[B_j] - \mathbb{E}[I_j])$ iteratively to ease computation. Expected overtime is derived from the observation that the completion time of client n, $C_n = \sum_{i=1}^{n-1} x_i + S_n$. When $\sum_{i=1}^{n-1} x_i > \mathrm{T}$, the last client arrives after the session-end time, in which case

$$\mathbb{E}[O] = \sum_{i=1}^{n-1} x_i + \mathbb{E}[S_n] - \mathrm{T}, \tag{2.17}$$

while in the other case $\sum_{i=1}^{n-1} x_i \leq \mathrm{T}$, so that

$$\begin{aligned}\mathbb{E}[O] &= \int_{\mathrm{T}-\sum_{i=1}^{n-1} x_i}^{\infty} \left(s - \left(\mathrm{T} - \sum_{i=1}^{n-1} x_i\right)\right) f_{S_n}(s)ds \\ &= m_{S_n}\left(\mathrm{T} - \sum_{i=1}^{n-1} x_i, 1\right) \\ &\quad - \left(\mathrm{T} - \sum_{i=1}^{n-1} x_i\right) \cdot m_{S_n}\left(\mathrm{T} - \sum_{i} x_i, 0\right). \end{aligned} \tag{2.18}$$

So, applying the approach to log-normal service times, we approximate each sojourn time S_i with a log-normal distribution, analogous to the Fenton approximation (Fenton 1960). As waiting times W_i do not follow a log-normal distribution, one might be concerned that is not a valid approximation; this, along with an extension to other distributions, is briefly discussed next. To help the reader, we provide an overview of the notation that we use in Table 2.1.

	Decision Variables	
$\boldsymbol{x}^*$	–	Optimal sequential solution.
$\bar{\boldsymbol{x}}$	–	Optimal simultaneous solution.
$\boldsymbol{x}^{*\texttt{MIM}}$, $\bar{\boldsymbol{x}}^{\texttt{MIM}}$	–	Optimal solutions found via the Moment-Iteration Method.
$\boldsymbol{x}^{*\texttt{SIM}}$, $\bar{\boldsymbol{x}}^{\texttt{SIM}}$	–	Optimal solutions found via simulation optimization.
	Cost Functions	
$\mathscr{F}_i(x_i\,;\omega)$	–	Sequential cost function for client i.
$\mathscr{F}(\boldsymbol{x};\omega)$	–	Simultaneous cost function without overtime.
$\mathscr{F}(\boldsymbol{x};\omega,\gamma)$	–	Simultaneous cost function with overtime.

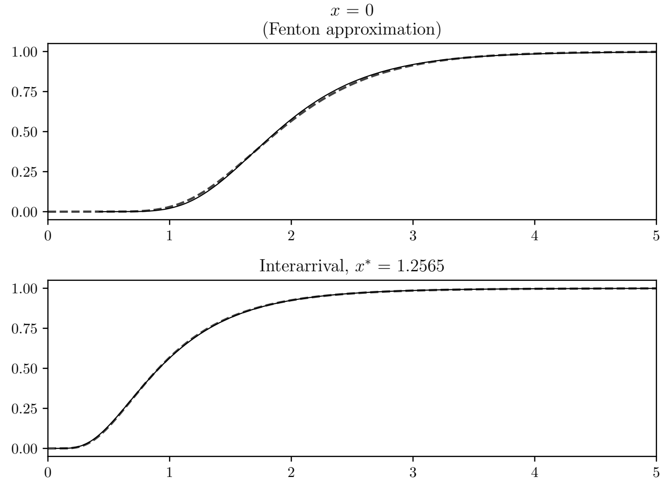
Table 2.1: Select notation.

2.4.1 Empirical Grounding

Consider a case with two clients. Suppose that the first client has a log-normally distributed service time B_1 with parameters μ_1 and σ_1, and CDF F_1. After time x the second client arrives, with log-normally distributed service time B_2 with parameters μ_2 and σ_2, and CDF F_2. Let W_2 denote the waiting time of the second client, and S_2 his or her sojourn time. As $W_2 = \max\{0, B_1 - x\}$ we have that the second client sees waiting time zero with probability $p = F_1(x)$, and with probability $1 - p$ the second client experiences positive waiting time. We will now describe the pdf of W_2 conditional upon $W_2 > 0$, which we abbreviate as $W_2|_{W_2>0}$,

$$f_{W_2|_{W_2>0}}(t) = \frac{\frac{1}{(t+x)\sigma_1\sqrt{2\pi}} e^{\frac{-(\log(t+x)-\mu_1)^2}{2\sigma_1^2}}}{1 - F_1(x)}.$$

Note that this is very similar to the three-parameter log-normal distribution as given in Kleiber and Kotz (2003). With probability p the second client's sojourn time distribution will simply equal that of B_2, which is log-normally distributed, and with probability $1 - p$ it will be the convolution of the distributions of $W_2|_{W_2>0}$ and B_2. Let us denote the pdf of such a distribution by $f_{S_2}(t) = pf_{B_2}(t) + (1-p)f_{S_2|_{W_2>0}}(t)$. In line with the well known Fenton approximation, to approximate the distribution given by f_{S_2} we consider the log-normal density. To scrutinize this choice, we simulated the above two-client example and compared the resulting empirical CDF to a log-normal distribution fitted via the method described in Section 2.5.1.


$x = 0$
(Fenton approximation)

Interarrival, $x^* = 1.2565$

Figure 2.1: The empirical CDF of S_2 found by simulation (solid line) compared to a log-normal CDF fitted by the first two moments (dashed line). In this example, the two distributions were differed by letting the first client have a standard deviation of 0.35 and the second 0.6.

In these examples, both clients were given a mean service time of one. We let the second client have a standard deviation of 0.6 (in the middle of our range), and let the first client have either a low standard deviation (0.35, in Figure 2.1) or a high standard deviation (0.85, in Figure 2.2). In both cases, assuming a weight on idle time of 0.8, we optimize the first inter-arrival time (only), i.e. minimizing $\mathscr{F}_1(x_1; \omega)$ of Eq. (2.2).

The reason behind the high idle weight of 80% is to encourage a large inter-arrival time and to understand how well the approximation works in such situations. In both cases we compare the optimal inter-arrival to $x = 0$ (i.e., simply the convolution of two log-normal distributions) to see how well the fit compares to the Fenton approximation. Both cases that we show are cases to which we might expect our approach to be applied, and in both cases we find a good fit. Therefore, we conclude that we can reasonably approximate the sojourn time distribution via a log-normal distribution, given that the service times are themselves log-normally distributed. Repeating this process for each client completes the argument that the moment-iteration moment works as an approximation for each client's sojourn time distribution, which is needed in Eq. (2.9).

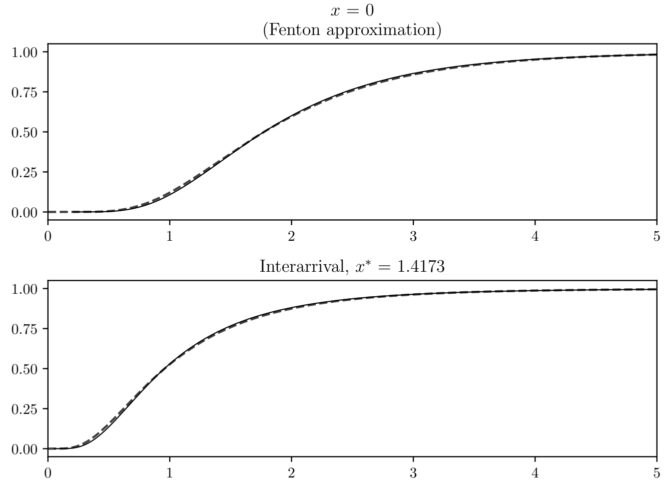


Figure 2.2: The empirical CDF of S found by simulation (solid line) compared to a log-normal CDF fitted by the first two moments (dashed line). In this example, the two distributions were differed by letting the first client have a standard deviation of 0.85 and the second 0.6.

2.4.2 Scheduling Multiple Clients

Now that we have established the goodness of the approximation at the scale of two subsequent clients, the question remains to be asked how well the moment-iteration method performs in the optimization of an entire appointment schedule. We will show how the method performs compared to *simulation* for the simultaneous optimization problems for a range of coefficients of variation and idle-weights: $\mathrm{cv} \in \{0.35, 0.85\}$ as in Çayırlı and Veral (2003) and $\omega \in \{0.5, 0.8\}$ as this coincides with our use cases.

The simultaneous optimization problem was solved under simulation by means of Sample Average Approximation (Kim et al. 2015). The solution was found by averaging the solutions to 100 runs, each with 20 000 replications. For each run a 'sample-optimum' was found using `L-BFGS-B` (Zhu et al. 1997) and the final estimator $\bar{\boldsymbol{x}}^{\texttt{SIM}}$ was found as the average over these sample-optima. This approach also allows us to estimate $\bar{\boldsymbol{x}}^{\texttt{SIM}}_{LCB}$ and $\bar{\boldsymbol{x}}^{\texttt{SIM}}_{UCB}$, the lower and upper 95% confidence bounds for the solution. The simulation was programmed in `Python`, using the package `numba` to just-in-time compile the main body of the simulation, which we call the `Inner Loop`. The package `scipy` was used for its implementation of `L-BFGS-B`. The moment-iteration method was likewise programmed in `Python` and likewise solved with `L-BFGS-B` for the simultaneous case. The simulation was significantly slower than the moment-iteration method to run, though the simulation would doubtless run faster in a compiled language, and may benefit from variance reduction techniques.

Firstly, we find in Tables 2.2 and 2.3 that the optimal solutions found by relying on the moment-iteration method $\bar{\boldsymbol{x}}^{\texttt{MIM}}$ also exhibit the typical dome-shape pattern found in optimal appointment scheduling. Next, we report for each setting an optimality gap by means of out-of-sample testing. We presume the simulation to be (arbitrarily close to) the true optimum and define the optimality gap as $(\mathscr{F}(\bar{\boldsymbol{x}}^{\texttt{MIM}}) - \mathscr{F}(\bar{\boldsymbol{x}}^{\texttt{SIM}}))/\mathscr{F}(\bar{\boldsymbol{x}}^{\texttt{SIM}})$. Again, we run 100 runs each of 20 000 replications, yielding over each run of 20 000 replications estimates for $\mathscr{F}(\bar{\boldsymbol{x}}^{\texttt{SIM}})$ and $\mathscr{F}(\bar{\boldsymbol{x}}^{\texttt{MIM}})$ (using the same random variables in each case) and over the 100 runs an estimate for the optimality gap.

The point estimators for the optimality gaps range from 0.0051% to 0.5921%. The reader may see this optimality gap of 0.5921% as being excessively large, but even in settings with high variance the moment-iteration method has merit. Firstly, as the number of clients grows, the moment-iteration-method allows one to calculate a schedule extremely quickly. Using sample average approximation, the increase in the number of clients leads to an increase of variance in the simulation, requiring many more runs to achieve good estimates. Secondly, it enables one to compare multiple different settings, such as a range of estimated coefficients of variation or weights placed on idle time, in a matter of seconds rather than hours. And thirdly, if solutions are to be rounded to the nearest minute or five minutes in practice, then the difference between the solutions given by simulation and the moment-iteration method disappear.

We now report the procedures for simultaneous optimization. We first report the simulation procedure as this is the approach most likely familiar to the reader. The procedure consists of two parts, an `Inner Loop` given in Algorithm 1 below, which, for a given vector of interarrivals $\boldsymbol{x}$, calculates es-

	CV = 0.35							
	$\omega = 0.5$				$\omega = 0.8$			
i	$\bar{x}^{\texttt{SIM}}_{i,LCB}$	$\bar{x}^{\texttt{SIM}}_{i,UCB}$	$\bar{x}^{\texttt{SIM}}_{i}$	$\bar{x}^{\texttt{MIM}}_{i}$	$\bar{x}^{\texttt{SIM}}_{i,LCB}$	$\bar{x}^{\texttt{SIM}}_{i,UCB}$	$\bar{x}^{\texttt{SIM}}_{i}$	$\bar{x}^{\texttt{MIM}}_{i}$
1	1.0467	1.0483	1.0475	1.0477	0.8064	0.8077	0.8071	0.8069
2	1.1894	1.1912	1.1903	1.1867	1.0162	1.0180	1.0171	1.0116
3	1.2104	1.2124	1.2114	1.2111	1.0518	1.0539	1.0529	1.0432
4	1.2170	1.2190	1.2180	1.2199	1.0645	1.0663	1.0654	1.0527
5	1.2173	1.2194	1.2184	1.2223	1.0654	1.0675	1.0664	1.0525
6	1.2167	1.2187	1.2177	1.2201	1.0637	1.0657	1.0647	1.0463
7	1.2089	1.2106	1.2097	1.2123	1.0530	1.0547	1.0538	1.0358
8	1.1954	1.1973	1.1963	1.1951	1.0350	1.0369	1.0359	1.0225
9	1.1659	1.1674	1.1666	1.1613	1.0025	1.0040	1.0033	1.0051
10	1.0799	1.0814	1.0807	1.0915	0.9250	0.9268	0.9259	0.9561
	Runtime		Opt. Gap		Runtime		Opt. Gap	
	MIM: 0.86s SIM: 671s		0.0051%		MIM: 1.02s SIM: 939s		0.1827%	

Table 2.2: A comparison of simulation and the moment-iteration method for simultaneously optimized schedules. 100 runs each of 20000 replications. Average runtime of a single simulation run given $\omega = 0.5$ was 5.54 seconds based on the last 50 runs, and was 10.28 seconds for $\omega = 0.8$.

timates of $\sum_{i=1}^{n} \mathbb{E}[I_i]$ and $\sum_{i=1}^{n} \mathbb{E}[W_i]$ and returns an objective value for, in our case, 20 000 sample "days", and an `Outer Loop` given in Algorithm 2, which optimizes 100 such repetitions and gives as the estimate of the optimum $\bar{\boldsymbol{x}}^{\texttt{SIM}} = \frac{1}{100}\sum_{k=1}^{100} \bar{\boldsymbol{x}}^{(k)}$. For each iteration k of the `Outer Loop` random variables $B_i^{(r)}$, $r = 1, \ldots, 20000$, $i = 1, \ldots, n$ are generated for which an optimum is found. That is we, do not generate new random variables each time an iteration of the minimizer is called, but rather find a minimum over a "frozen" sample of 20 000 days, which is repeated by the `Outer Loop` 100 times, each with a new sample, to yield an estimate for the optimal solution.

Turning our attention to the Moment-Iteration Method, the pseudo-code given in Algorithm 3 relates how to structure the function that should be passed to an optimizer. $\mathbb{E}[W_i]$ is calculated as in Eq. (2.14), $\mathbb{E}[I_i]$ as in Eq. (2.16), and one can include $\mathbb{E}[O]$ via (2.17) and (2.18), if relevant. This algorithm requires no `Outer Loop` as it can be called directly with a minimizer.

2.5 Sequentially Optimal Schedules via the Moment Iteration Method

Next, we present the algorithm that combines sequential optimization with the moment-iteration method in the exponential and log-normal cases. Note that this can be extended to other distributions, for example, location-scale families in which many of the sequencing results which we will show later also hold, or even distribution-free (Gallego and Moon 1993), as after all the approach

Algorithm 1 `Inner Loop` for Simulation

```
Given a vector of inter-arrivals $\boldsymbol{x}$ calculate arrival epochs $t_i = \sum_{j=1}^{i} x_i$
Given $B_i^{(r)} \sim \mathcal{LN}(\nu, \tau^2)$, $r = 1, \ldots, 20000$, $i = 1, \ldots, n$
Initialize $EI,\ EW = 0$    ▷ Estimators of $\sum_i \mathbb{E}[I]$ and $\sum_i \mathbb{E}[W]$
for $r = 1, 2, \ldots, 20000$ do
    Initialize $\mathcal{I},\ \mathcal{W}$, time $= 0$  ▷ $\mathcal{I}$ and $\mathcal{W}$, sum of observed idle and wait times.
    for $i = 2, 3, \ldots, n$ do
        if time $\leq t_i$ then
            $\mathcal{I} = \mathcal{I} + t_i -$ time
            time $= t_i + B_i^{(r)}$
        else
            $\mathcal{W} = \mathcal{W} +$ time $- t_i$
            time $=$ time $+ B_i^{(r)}$
        end if
    end for
    $EI = EI + (\mathcal{I} - EI)/r$    ▷ Update estimators of $\mathbb{E}[I]$ and $\mathbb{E}[W]$
    $EW = EW + (\mathcal{W} - EW)/r$
end for
Return $\mathscr{F} = \omega EI + (1 - \omega) EW$
```

Algorithm 2 `Outer Loop` for Simulation

```
for $k = 1, 2, \ldots, 100$ do
    Draw $\boldsymbol{B} = B_i^{(r)} \sim \mathcal{LN}(\nu, \tau^2)$, $r = 1, \ldots, 20000$, $i = 1, \ldots, n$
    Return $\bar{\boldsymbol{x}}^{(k)} = \arg\min(\texttt{Inner Loop}(\boldsymbol{B}))$    ▷ This paper uses L-BFGS-B
end for
Return $\bar{\boldsymbol{x}}^{\texttt{SIM}} = \frac{1}{100} \sum_{k=1}^{100} \bar{\boldsymbol{x}}^{(k)}$
```

Algorithm 3 Procedure for Simultaneous Scheduling via the Moment-Iteration Method (Log-Normal)

```
Given a vector $\boldsymbol{x} = (x_i)_{i=1}^{n-1}$ found by an iteration of an optimization algorithm of your choice
$S_1 = B_1 \sim \mathcal{LN}(\nu_1, \tau_1^2)$
for $i = 2, 3, \ldots, n-1$ do
    Calculate expected sojourn time: $\mathbb{E}[S_i] = \mathbb{E}[B_i] + \mathbb{E}[W_i]$
    Calculate variance of sojourn time: $\mathbb{V}\text{ar}[S_i] = \mathbb{V}\text{ar}[B_i] + \mathbb{V}\text{ar}[W_i]$
    Fit $\nu_i = \log\left(\frac{\mathbb{E}[S_i]^2}{\sqrt{\mathbb{V}\text{ar}[S_i] + \mathbb{E}[S_i]^2}}\right)$
    Fit $\tau_i^2 = \log\left(1 + \frac{\mathbb{V}\text{ar}[S_i]}{\mathbb{E}[S_i]^2}\right)$
    Fit $S_i \sim \mathcal{LN}(\nu_i, \tau_i^2)$
    Calculate expected idle and waiting times: $\mathbb{E}[I_i]$ and $\mathbb{E}[W_i]$
end for
Return $\mathscr{F} = \omega \sum_{i=1}^{n} \mathbb{E}[I_i] + (1-\omega) \sum_{i=1}^{n} \mathbb{E}[W_i]$
```

Algorithm 4 Procedure for Sequential Scheduling via the Moment-Iteration Method (Log-Normal)

```
$S_1 = B_1 \sim \mathcal{LN}(\nu_1, \tau_1^2)$
Find optimal inter-arrival time $x_1^* = \mathrm{e}^{\mu_1 + \sigma_1 \Phi^{-1}(1-\omega)}$
for $i = 2, 3, \ldots, n-1$ do
    Calculate expected sojourn time $\mathbb{E}[S_i] = \mathbb{E}[B_i] + \mathbb{E}[W_i]$
    Calculate variance of sojourn time $\mathbb{V}\text{ar}[S_i] = \mathbb{V}\text{ar}[B_i] + \mathbb{V}\text{ar}[W_i]$
    Fit $\nu_i = \log\left(\frac{\mathbb{E}[S_i]^2}{\sqrt{\mathbb{V}\text{ar}[S_i] + \mathbb{E}[S_i]^2}}\right)$
    Fit $\tau_i^2 = \log\left(1 + \frac{\mathbb{V}\text{ar}[S_i]}{\mathbb{E}[S_i]^2}\right)$
    Fit $S_i \sim \mathcal{LN}(\nu_i, \tau_i^2)$
    Find $x_i^* = \mathrm{e}^{\nu_i + \tau_i \Phi^{-1}(1-\omega)}$
end for
```

	CV = 0.85							
	$\omega = 0.5$				$\omega = 0.8$			
i	$\bar{x}^{\texttt{SIM}}_{i,LCB}$	$\bar{x}^{\texttt{SIM}}_{i,UCB}$	$\bar{x}^{\texttt{SIM}}_{i}$	$\bar{x}^{\texttt{MIM}}_{i}$	$\bar{x}^{\texttt{SIM}}_{i,LCB}$	$\bar{x}^{\texttt{SIM}}_{i,UCB}$	$\bar{x}^{\texttt{SIM}}_{i}$	$\bar{x}^{\texttt{MIM}}_{i}$
1	0.9860	0.9899	0.9879	0.9962	0.5457	0.5478	0.5468	0.5369
2	1.3733	1.3783	1.3758	1.3469	0.9457	0.9492	0.9474	0.9095
3	1.4576	1.4632	1.4604	1.4138	1.0554	1.0598	1.0576	1.0035
4	1.4895	1.4950	1.4922	1.4353	1.0993	1.1036	1.1015	1.0334
5	1.4946	1.5003	1.4975	1.4367	1.1080	1.1131	1.1106	1.0451
6	1.4933	1.4991	1.4962	1.4236	1.1072	1.1120	1.1096	1.0488
7	1.4678	1.4736	1.4707	1.3988	1.0810	1.0851	1.0831	1.0462
8	1.4220	1.4270	1.4245	1.3624	1.0352	1.0391	1.0372	1.0334
9	1.3266	1.3303	1.3284	1.3010	0.9552	0.9583	0.9567	0.9908
10	1.0788	1.0819	1.0803	1.1133	0.7759	0.7790	0.7775	0.8511
	Runtime		Opt. Gap		Runtime		Opt. Gap	
	MIM: 0.51s SIM: 749s		0.3695%		MIM: 0.66s SIM: 700s		0.5921%	

Table 2.3: A comparison of simulation and the moment-iteration method for simultaneously optimized schedules. 100 runs each of 20000 replications. Average runtime of a single simulation run given $\omega = 0.5$ was 7.82 seconds based on the last 70 runs, and was 8.03 seconds for $\omega = 0.8$.

reduces the problem to a series of newsvendor problems. The code is simple to implement in many programming languages to generate fast and accurate sequentially optimized schedules.

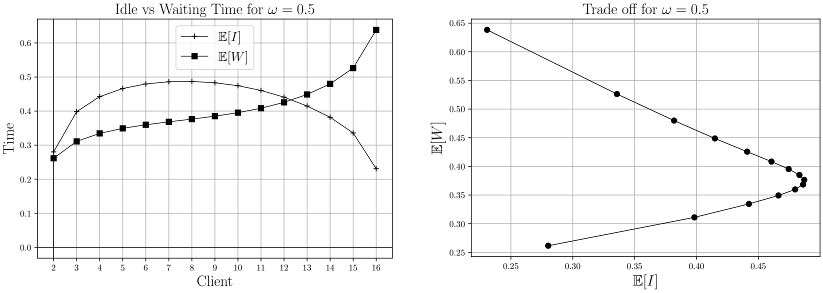
2.5.1 Log-Normal

Assuming log-normally distributed service times, the resulting optimal inter-arrival times that minimize the objective function defined in Equation (2.2) are found by

$$x_i^* = \mathrm{e}^{\nu_i + \tau_i \Phi^{-1}(1-\omega)}, \tag{2.19}$$

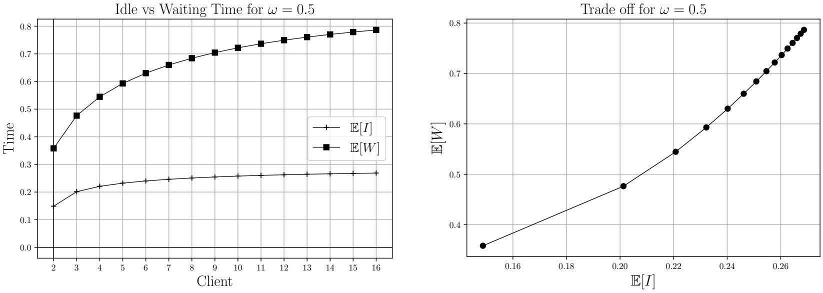
where the parameters are given in Eq. (2.12). The technically most complicated requirement is that values for the inverse normal distribution are available for calculation of x_i^* as per Eq. (2.19). This approach is given in Algorithm 4. Steps 4 through 8 can be used to apply the moment-iteration method to other optimization approaches.

In Figure 2.3 we present the simultaneous schedule for 16 clients, each with mean $\mathbb{E}[B] = 1$ and coefficient of variation $\mathrm{cv}_B = 0.6$. We plot the expected idle and waiting times, $\mathbb{E}[I_i]$, $\mathbb{E}[W_i]$, for clients $i \geq 2$. Note how expected waiting time increases sharply for clients at the end of the schedule in simultaneous optimization (Figure 2.3a). This is as a schedule overrun affects fewer clients and carries less weight than the opposite ambition of reducing expected idle time. As a result, we see considerably different cost ratios for clients at the beginning versus the end of the session, making it difficult to determine the effect of a particular choice of weight parameter. Figure 2.3b depicts this same



(a) Waiting and idle times per client. (b) Waiting versus idle time.

Figure 2.3: Simultaneous optimization and how expected waiting time behaves compared to expected idle time. Here we are looking at a schedule for 16 clients with $\omega = 0.5$.



(a) Waiting and idle times per client. (b) Waiting versus idle time.

Figure 2.4: Sequential optimization and how expected waiting time behaves compared to expected idle time. Here we are looking at a schedule for 16 clients with $\omega = 0.5$.

result by plotting each client's expected waiting time against his or her expected idle time.

In Figure 2.4, we depict a sequentially optimized schedule. We see how the expected idle and waiting times grow proportionally with each other throughout the schedule. In Figure 2.4b, we also see how the trade off between expected waiting and idle time seems to tend towards a steady-state value. In this way, a particular value of ω can be paired with a long-run cost ratio. There is an unavoidable trade-off between clients' waiting time and the physician's idle time, and sequential schedules have the advantage that this trade-off per client is more transparent. In Section 2.7 we present simultaneous and sequential schedules in more depth for the reader to compare.

This approach is by no means restricted to the log-normal distribution. Any distribution for which the partial moments can be calculated can be employed, although verification as described earlier in this section should be carried out. To illustrate, we extend the approach to the exponential distribution, and also provide the partial moments for the Weibull and gamma distributions in Section 2.5.3. Furthermore, it can be extended to a distribution-free setting, which is treated in Section 2.8.

2.5.2 Exponential

We now illustrate how to extend application of the moment-iteration method in combination with the sequential heuristic method to the exponential distribution. In the case that the service times of the clients are distributed according to an exponential distribution, we have that $B_i \sim \text{Exp}(\lambda_i)$ for $i = 1, \dots, n$. As the exponential distribution depends on only one parameter, we choose this parameter μ_i ($S_i \sim \text{Exp}(\mu_i)$) to ensure that the expected sojourn time of the next client matches the mean waiting and service time, rather than focusing on its second moment, i.e.,

$$\mathbb{E}[S_i] = \mathbb{E}[B_i] + \mathbb{E}[W_i] = \frac{1}{\lambda_i} + \frac{\omega}{\mu_{i-1}}, \tag{2.20}$$

where the waiting time $\mathbb{E}[W_i] = \mathbb{E}[(S_{i-1} - x_{i-1})^+]$ follows from the sequential solution of (2.5) for the previous client:

$$x^*_{i-1} = -\frac{\log(\omega)}{\mu_{i-1}}. \tag{2.21}$$

So, the application of the moment-iteration method in combination with the sequential heuristic method of Section 2.3 results in the following iterative scheme, note that $\mu_1 = \lambda_1$. Based on this scheme, a sequencing rule can be derived that minimizes the objective sequentially, see Proposition 2.3.

Algorithm 5 Procedure for Sequential Scheduling via the Moment-Iteration Method (Exponential)

```
$S_1 \overset{\mathcal{D}}{=} B_1 \sim \text{Exp}(\lambda_1)$
Find optimal inter-arrival time $x^*_1 = -\frac{\log(\omega)}{\mu_1}$
for $i = 2, 3, \dots, n-1$ do
    Calculate expected sojourn time $\mathbb{E}[S_i] = \mathbb{E}[B_i] + \mathbb{E}[W_i]$
    Fit $\mu_i = \frac{1}{\mathbb{E}[S_i]}$
    Fit $S_i \sim \text{Exp}(\mu_i)$
    Find $x^*_i = -\frac{\log(\omega)}{\mu_i}$
end for
```

2.5.3 Weibull and Gamma

For the Weibull distribution with probability density function $f(s) = \frac{\alpha}{\lambda}\left(\frac{x}{\lambda}\right)^{\alpha-1} e^{-(x/\lambda)^\alpha}$ we arrive at the kth partial moment $m_S(x,k) = \lambda^k \Gamma\left(\frac{k}{\alpha}+1, \frac{k}{\alpha}(x/\lambda)^\alpha\right)$ by a change of variables $t = (s/\lambda)^\alpha$. While for the gamma distribution with density function $f(s) = \frac{1}{\Gamma(\alpha)\lambda^{\alpha}} e^{-s/\lambda}$ we arrive at the kth partial moment, $m_S(x,k) = \frac{\lambda^k}{\Gamma(\alpha)}\Gamma(k+\alpha, x/\lambda)$ by integrating $\int_x^\infty s^k f(s) ds$ with a change of variables $t = s/\lambda$. Both of these expressions contain the upper incomplete gamma function, $\Gamma(\alpha, x) = \int_x^\infty s^{\alpha-1} e^{-s} ds$, which can be speedily retrieved from many numerical packages, such as `scipy.special.gammaincc` (indeed with two c-s) and `Matlab`'s `gammainc` as well as via `ALGLIB` and `GSL` for `C/C++` and `Apache Commons` for `Java`.

2.6 Sequencing of Clients

Given heterogeneous clients' service times, besides the question *when* to schedule, the question of *whom* to schedule also comes into play. Consider, for example, the omnipresent distinction between *new* and *return* clients in healthcare (e.g. Çayırlı et al. 2006). Even with two types of clients, the complexity of the sequencing problem increases rapidly: if there are `N` new and `R` return clients then there are $\binom{\mathtt{N}+\mathtt{R}}{\mathtt{N}}$ unique sequences in which clients can be scheduled, each requiring an optimization over the (inter-)arrival times.

The difficulty of this problem can be greatly reduced by integrating the two approaches presented in Sections 2.4 and 2.3. Under the sequential scheduling paradigm, i.e., sequential optimization, we only have to consider one client at a time. At the ith client to be scheduled, the previous arrivals are already scheduled and do not interfere with the sequencing decision, which is formalized in the next proposition. This results in the sequencing and scheduling of the next client such that the resulting schedule is *sequentially optimal*, i.e., it minimizes the additional costs $\mathscr{F}_i(x_i;\omega)$ iteratively.

Proposition 2.2. *Consider a set of clients* $1, 2, \ldots, i-1$, *for which a predetermined schedule exists. If client i will be appended to the schedule, then the choice of this client is irrespective of the past, i.e., the clients that are already scheduled, and impacts client* $i+1$*'s idle and waiting times only via his/her inter-arrival time* x_i.

Proof. Define for any ω the non-negative loss function $\ell_x(t) = \omega\,(x-t)^+ + (1-\omega)\,(t-x)^+$ where $(\cdot)^+ = \max\{\cdot, 0\}$, the expected loss for the ith client can be expressed as

$$\begin{aligned}\mathscr{F}_i(x_i;\omega) = \omega\,\mathbb{E}[I_{i+1}] + (1-\omega)\,\mathbb{E}[W_{i+1}] &= \int_0^\infty \ell_{x_i}(s) f_{S_i}(s)\,ds \\ &= \int_0^\infty \ell_{x_i}(s)\left(\int_0^\infty f_{B_i}(y) f_{W_i}(s-y)\;dy\right)ds, \\ &= \int_0^\infty \left(\int_0^\infty \ell_{x_i-t}(y) f_{B_i}(y)\,dy\right) f_{W_i}(t)\,dt, \qquad (2.22)\end{aligned}$$

where a change of variables is applied. Note for $i=1$ that $f_{W_1}(t)$ has mass 1 at $t=0$ and mass 0 elsewhere. Observe that the inner integral over the loss function is irrespective of the waiting-time distribution $f_{W_i}(t)$. Since previous inter-arrivals are already determined, we have a vector $\hat{x}_{[1:i-1]} = (\hat{x}_1, \ldots, \hat{x}_{i-1})$, so that:

$$\begin{aligned} W_i(\hat{x}_{[1:i-1]}) = \max\{0,\ & B_{i-1} - \hat{x}_{i-1}, \\ & B_{i-1} + B_{i-2} - (\hat{x}_{i-1} + \hat{x}_{i-2}), \ldots, \sum_{j=1}^{i-1}(B_{i-j} + \hat{x}_{i-j})\}. \end{aligned}$$

Thus, the density function $f_{W_i}(t)$ in Eq. (2.22) does not depend on x_i. Thereby, the decision who to schedule in the ith position does not depend on the $i-1$ clients scheduled earlier. □

Corollary 2.2.1. *In case of heterogeneous clients, there is a sequence of clients which is sequentially optimal.*

Proof. From Proposition 2.2, if i clients are scheduled, the ith sequencing decision is independent of the previous clients that are scheduled for *any* scheduling policy. Hence, from the $n-i$ clients remaining, the client should be selected that brings about the lowest additional costs, which is obtained via the sequential optimization procedure, i.e., the client with the lowest $\mathscr{F}_i(x_i;\omega)$. □

Determining the next (sequentially) optimal inter-arrival still requires opaque convolutions. Luckily, with the moment-iteration method of Section 2.4 the next optimal inter-arrival time is readily found. Moreover, relying on this method one also knows which distribution is used to model the sojourn time from which the partial costs $\mathscr{F}_i(\boldsymbol{x}^*;\omega)$ are obtained. We show that for sequential schedules combined with the moment-iteration method, comprehensive sequencing rules can be found based on the mean, variance and *combinations* of both.

Before we substantiate the log-normal case, we briefly handle the exponential case as its sequencing question also received significant interest (Kaandorp and Koole 2007, Wang 1999, Choi and Wilhelm 2020).

Proposition 2.3. *Using the moment-iteration method and sequential optimization in the case of exponential service times, sequencing lowest mean/variance first is sequentially optimal.*

Proof. We consider an arbitrary client $i > 1$, and for brevity we omit the subscript i. The objective function as defined in Eq. (2.2) can be written as

$$\begin{aligned} \mathscr{F}(x;\omega) &= \omega\mathbb{E}[(x-B)^+] + (1-\omega)\mathbb{E}[(B-x)^+] \\ &= \omega(x - \mathbb{E}[B]) + \mathbb{E}[W]. \end{aligned} \tag{2.23}$$

Filling in x^* of Eq. (2.21), we find:

$$\mathscr{F}(x^*;\omega) = \omega\left(x^* - \frac{1}{\lambda}\right) + \frac{\mathrm{e}^{-\lambda x^*}}{\lambda} = -\frac{\omega\log(\omega)}{\lambda} = -\omega\log(\omega)\mathbb{E}B. \tag{2.24}$$

This function is decreasing in λ. Hence, the client with the highest rate parameter λ (lowest mean/variance) is desired to minimize costs. By applying the Moment Iteration Method at each iteration, each problem of (2.23) becomes dependent on the exponential distribution with the parameter following from the expected sojourn time. This expectation is maximized when the client with highest mean/variance is chosen by Eq. (2.20). □

Within appointment scheduling, uncertainty in service times is passed on to subsequent clients, as the waiting-time distribution is part of the sojourn-time distribution via the relation of Eq. (2.9). This logic provides grounds for using the sequencing heuristic `SVF`, which is shown to be sequentially optimal in Proposition 2.3. However, in the previous case mean and variance are confounded via the single *scale* parameter characterizing the distribution. The case of log-normal service times is more involved as both the mean and variance are free to move in different directions. The machinery of the moment-iteration method prescribes to re-estimate according to Equations (2.10) and (2.11), which permits unravelling sequencing results in three directions:

- *Scale.* The means and standard deviation increase with the same magnitude, cf. Proposition 2.3.
- *Variance.* Keep the means constant and have increasing standard deviations.
- *Mean.* Keep the variances constant and have increasing means.

In addition, Section 2.7 offers contour plots for situations that might occur in practice in which our sequencing rules do not immediately provide an answer.

2.6.1 Sequencing by Standard Deviation

Even though the `SVF` rule has intuitive grounds, the simulations, e.g. in Çayırlı and Yang (2014), show that there is no universal rule that works best in practice. Specifically (Kong et al. 2016) show that, assuming log-normal service times in equidistant schedules based on service-time averages, the `SVF` rule is not optimal when there are more than 80 clients. For our sequential optimization approach as outlined in Section 2.3 we prove that, when means are held equal, clients should be scheduled in increasing order of variance.

Theorem 2.1. *For a log-normally distributed variable B, the expected idle and waiting times are increasing in variance, while the mean is kept constant, for all inter-arrival times $x > 0$.*

Proof. We consider an arbitrary client $i > 1$, for brevity we omit the index i. Note that the mean is $\mathbb{E}[B] = \mathrm{e}^{\mu+\sigma^2/2}$ and the squared coefficient of variation is $\mathrm{scv}_B = \mathrm{e}^{\sigma^2} - 1$, both of which are increasing in σ. Keeping the mean constant by setting $\mu = -\sigma^2/2 + \log \mathbb{E}[B]$ we will show that

$$\frac{\partial \mathbb{E}[I]}{\partial \sigma} \geq 0 \text{ and } \frac{\partial \mathbb{E}[W]}{\partial \sigma} \geq 0.$$

First, we rewrite $\mathbb{E}[I]$ and $\mathbb{E}[W]$

$$\mathbb{E}[I] = xF_B(x) - \int_0^x tf_B(t)\ dt \quad \text{and} \quad \mathbb{E}[W] = xF_B(x) - x + \int_x^\infty tf_B(t)dt.$$

Using the error function $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x \mathrm{e}^{-t^2}dt$ we find that

$$\int_a^b tf_B(t)dt = \\ \frac{\mathbb{E}[B]}{2}\left(\mathrm{erf}\left(\frac{\log(b/\mathbb{E}[B]) - \frac{\sigma^2}{2}}{\sqrt{2}\sigma}\right) - \mathrm{erf}\left(\frac{\log(a/\mathbb{E}[B]) - \frac{\sigma^2}{2}}{\sqrt{2}\sigma}\right)\right)$$

for $0 \leq a \leq b$, which can be found by a change of variables $u = \frac{\log t - \mu + \sigma^2}{\sqrt{2}\sigma}$. Combined with the definition

$$F_B(x) = \frac{1}{2}\mathrm{erf}\left(\frac{\log(x/\mathbb{E}[B]) + \frac{\sigma^2}{2}}{\sqrt{2}\sigma}\right) + \frac{1}{2},$$

this allows us to write

$$\begin{aligned} \mathbb{E}[I] &= \frac{x}{2}\left(\mathrm{erf}\left(y_+(\sigma)\right) + 1\right) - \frac{\mathbb{E}[B]}{2}\left(\mathrm{erf}\left(y_-(\sigma)\right) + 1\right), \\ \mathbb{E}[W] &= \frac{x}{2}\left(\mathrm{erf}\left(y_+(\sigma)\right) - 1\right) - \frac{\mathbb{E}[B]}{2}\left(\mathrm{erf}\left(y_-(\sigma)\right) - 1\right), \end{aligned} \tag{2.25}$$

where

$$y_+(\sigma) = \frac{\log(x/\mathbb{E}[B]) + \frac{\sigma^2}{2}}{\sqrt{2}\sigma} \quad \text{and} \quad y_-(\sigma) = \frac{\log(x/\mathbb{E}[B]) - \frac{\sigma^2}{2}}{\sqrt{2}\sigma}.$$

We are interested in

$$\frac{\partial}{\partial\sigma}\mathbb{E}[I] = \frac{\partial}{\partial\sigma}\mathbb{E}[W] = \frac{x}{2}\frac{\partial}{\partial\sigma}\mathrm{erf}\big(y_+(\sigma)\big) - \frac{\mathbb{E}[B]}{2}\frac{\partial}{\partial\sigma}\mathrm{erf}\big(y_-(\sigma)\big),$$

wherein

$$\frac{\partial}{\partial\sigma}\mathrm{erf}\big(y_+(\sigma)\big) = \frac{2}{\sqrt{\pi}}\mathrm{e}^{-y_+(\sigma)^2} \cdot \frac{\frac{\sigma^2}{2} - \log(x/\mathbb{E}[B])}{\sqrt{2}\sigma^2},$$

and

$$\frac{\partial}{\partial\sigma}\mathrm{erf}\big(y_-(\sigma)\big) = \frac{-2}{\sqrt{\pi}}\mathrm{e}^{-y_-(\sigma)^2} \cdot \frac{\frac{\sigma^2}{2} + \log(x/\mathbb{E}[B])}{\sqrt{2}\sigma^2}.$$

Filling in the expressions for $\frac{\partial y_\pm(\sigma)}{\partial\sigma}$ and cancelling out terms, we arrive at

$$\frac{\partial}{\partial\sigma}\mathbb{E}[I] = \frac{\partial}{\partial\sigma}\mathbb{E}[W] = \sqrt{\frac{x\mathbb{E}[B]}{2\pi}}\mathrm{e}^{-\frac{4\log(x/\mathbb{E}[B])^2 + \sigma^4}{8\sigma^2}} > 0.$$

□

Corollary 2.1.1. *Using the log-normal approximation method, the objective functions of Eq.* (2.2) *are sequentially minimized when lowest variance clients are scheduled first given that they have equal means.*

Proof. Consider an arbitrary client i. Without loss of generality define two service-time variables B_1 and B_2 where $\mathbb{E}[B_1] = \mathbb{E}[B_2]$, but $\mathbb{V}\text{ar}[B_1] \leq \mathbb{V}\text{ar}[B_2]$, then we have by Theorem 2.1 that

$$\begin{aligned}\mathscr{F}_i^{B_1}(x_i^*(B_1);\omega) &= \int_0^\infty \ell_{x_i^*(B_1)}(t) f_{B_1}(t)\ dt \\ &\leq \int_0^\infty \ell_{x_i^*(B_2)}(t) f_{B_1}(t)\ dt \leq \mathscr{F}_i^{B_2}(x_i^*(B_2);\omega),\end{aligned}$$

where $x_i^*(B_j)$ denotes the inter-arrival time which minimizes the objective function $\mathscr{F}_i^{B_j}(x_i;\omega)$ when B_j is taken as the service time of the ith client. Due to Proposition 2.2, the argument can be iteratively applied to minimize the objective function sequentially. □

Given Theorem 2.1, we are now aware how expected idle and waiting times increase in the variance of a log-normally distributed service time. By using the approximation from Section 2.4 of replacing the sojourn-time distribution by a single log-normal distribution that matches the first two moments, and by Proposition 2.2, it suffices to look only at the moments of the service time of the client to be scheduled next. This statement also holds for the following findings.

2.6.2 Sequencing by Mean and Standard Deviation

Now we proceed to vary the standard deviation as well as the mean. A first step to doing this is by fixing the squared coefficient of variation

$$\text{scv} = \frac{\mathbb{V}\text{ar}[B]}{\mathbb{E}[B]^2} = \mathrm{e}^{\sigma^2} - 1, \tag{2.26}$$

which is a function of σ. In addition, the following results rely on the fact that we plug in the optimal inter-arrival time x^*.

Proposition 2.4. *For a log-normally distributed variable B, if the mean and standard deviation increase with the same rate, then the expected idle and waiting times increase.*

Proof. We consider an arbitrary client $i > 1$, and for brevity we omit the subscript i. Since the mean and standard deviation increase with the same rate, the SCV is fixed. This means that by Eq. (2.26) σ also remains constant, and μ increases.

Rewriting the waiting time by using Eq. (2.13) and plugging in x^* of Eq. (2.19) results in

$$\begin{aligned}\mathbb{E}[W] &= m(x^*,1) - x^* m(x^*,0)\\ &= \mathbb{E}[B]\Phi\left(-\Phi^{-1}(1-\omega)+\sigma\right) - x^*\Phi(-\Phi^{-1}(1-\omega))\\ &= \mathrm{e}^{\mu}\left(\Phi\left(\sigma+\Phi^{-1}(\omega)\right)\mathrm{e}^{\frac{1}{2}\sigma^2} - \omega\,\mathrm{e}^{\sigma\Phi^{-1}(1-\omega)}\right), \qquad (2.27)\end{aligned}$$

where we used that $-\Phi^{-1}(1-\omega) = \Phi(\omega)$. Taking the derivative in μ shows that the waiting time increases when μ increases. Furthermore, the idle time considered in the optimum x^* can be expressed as

$$\begin{aligned}\mathbb{E}[I] &= x^* - \mathbb{E}[B] - \mathbb{E}[W]\\ &= \mathrm{e}^{\mu}\left(\mathrm{e}^{\sigma\Phi^{-1}(1-\omega)}(1+\omega) - \left(1+\Phi\left(\sigma+\Phi^{-1}(\omega)\right)\right)\mathrm{e}^{\frac{1}{2}\sigma^2}\right),\end{aligned}$$

which obviously increases in μ as well. □

From Proposition 2.4 it follows that if the mean increases by α then the expected idle and waiting times, and thus the objective function, increase by the same factor. This is no surprise, as the log-normal distribution is a *scale family* and apparently this property carries over nicely to idle and waiting times. Combined with the approximation from Section 2.4 and with Proposition 2.2, we know that, when minimizing the schedule sequentially, clients with a lower CV should precede clients with a higher CV, as long as the mean is non-decreasing; see Figure 2.5 for a sketch of the situation.

Example 2.1. *Typically in healthcare, appointments for new clients require more time and have relatively more variation than for return clients. An instance that fits these characteristics is found in Çayırlı et al. (2006), where new clients have longer service times and more variation compared to return clients;* CV *0.360 vs. 0.325 and mean 19.09 vs. 15.50 minutes. The sequential optimization approach as well as the sequencing rule can therefore be used to generate a schedule for a given number of new and return clients.*

2.6.3 Sequencing by Mean

We focus on the case where we keep the standard deviation fixed while changing the mean, so that we move in the horizontal direction in Figure 2.5. Interestingly, an increase in the mean can only be guaranteed to have an increasing effect on the objective function when ω is chosen to be at least one half.

Theorem 2.2. *For a log-normally distributed variable B and $\omega \geq 0.5$, the minimized objective function is increasing in the mean, while standard deviation is kept constant.*

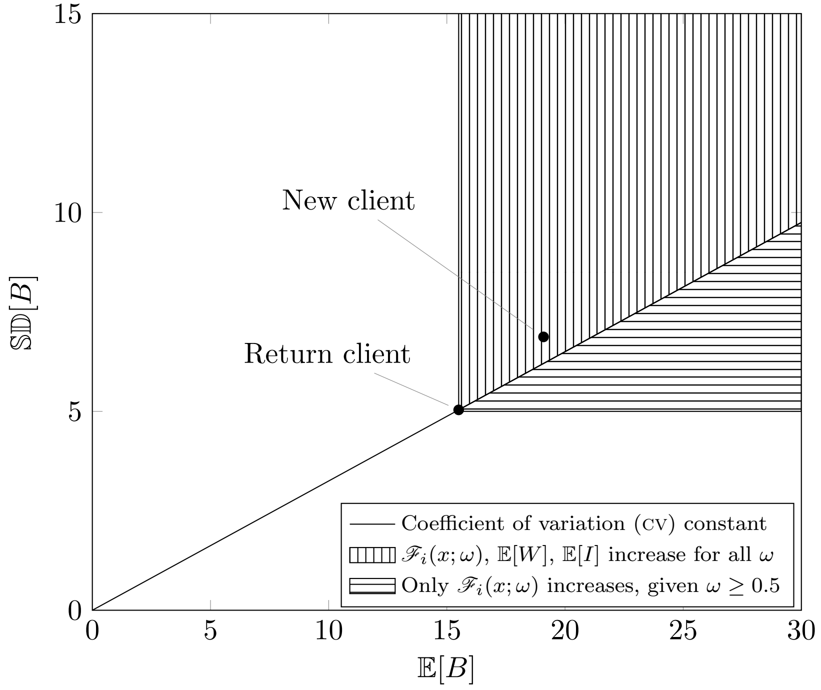


Figure 2.5: Situation sketch. Along the horizontal axis we denote the mean and along the vertical the standard deviation. The service time distributions for both a return and a new client as in Example 2.1 have been plotted. The diagonal where CV is fixed delineates two regions. Above the diagonal, the expected idle and waiting times increase; whereas in the cone below the diagonal, it is known that only the objective function will increase as long as $\omega \geq 0.5$.

Proof. Using Eq. (2.27) we obtain

$$\mathscr{F}(x^*;\omega) = \left(\Phi\left(\sigma + \Phi^{-1}(\omega)\right) - \omega\right) \mathrm{e}^{\mu + \frac{1}{2}\sigma^2}. \tag{2.28}$$

Writing Eq. (2.28) out in mean and variance, we find that

$$\mathscr{F}(x^*;\omega) = \left(\Phi\left(\sqrt{\log\frac{\mathbb{E}[B]^2 + \mathbb{V}\mathrm{ar}\,[B]}{\mathbb{E}[B]^2}} + \Phi^{-1}(\omega)\right) - \omega\right)\mathbb{E}[B].$$

So that, when taking the derivative in the mean $\mathbb{E}[B]$, we arrive at

$$\begin{aligned}\frac{\partial\mathscr{F}(x^*;\omega)}{\partial\mathbb{E}[B]} &= \left(\Phi\left(\sigma + \Phi^{-1}(\omega)\right) - \omega\right) \\ &\qquad - \frac{\mathbb{V}\mathrm{ar}\,[B]}{\mathbb{E}[B]^2 + \mathbb{V}\mathrm{ar}\,[B]}\frac{1}{\sigma}\varphi\left(\Phi\left(\sigma + \Phi^{-1}(\omega)\right)\right) \\ &= \Phi\left(\Phi^{-1}(\omega) + \sigma\right) - \Phi\left(\Phi^{-1}(\omega)\right) - \frac{1 - e^{-\sigma^2}}{\sigma}\varphi\left(\Phi^{-1}(\omega) + \sigma\right). \end{aligned} \tag{2.29}$$

Using the fact that $\omega \geq 0.5$ we note that the function $\Phi(x)$ is concave for $x \geq 0$. Now, let f be a function, concave for all $x \geq a - b$, with $b \geq 0$:

$$f(a) - f(a-b) \geq b \cdot f'(a),$$

i.e., the tangent at $a = \Phi^{-1}(\omega) + \sigma$ lies above the CDF at $a - \sigma = \Phi^{-1}(\omega)$, and so $\Phi\left(\Phi^{-1}(\omega) + \sigma\right) - \Phi\left(\Phi^{-1}(\omega)\right)$ is at least as great as $\sigma\varphi\left(\Phi^{-1}(\omega)\right)$, and strictly greater when $\sigma > 0$, so that

$$\begin{aligned}\frac{\partial\mathscr{F}(x^*;\omega)}{\partial\mathbb{E}[B]} &\geq \sigma\varphi\left(\Phi^{-1}(\omega) + \sigma\right) - \frac{1 - e^{-\sigma^2}}{\sigma}\varphi\left(\Phi^{-1}(\omega) + \sigma\right) \\ &= \left(\sigma - \frac{1 - e^{-\sigma^2}}{\sigma}\right)\varphi\left(\Phi^{-1}(\omega) + \sigma\right). \end{aligned} \tag{2.30}$$

For this to be positive, $\sigma^2 + e^{-\sigma^2}$ must be greater than 1. Let $h(\sigma) = \sigma^2 + e^{-\sigma^2}$, then $h(0) = 1$ and its derivative is non-negative, thus Eq. (2.30) is always non-negative, and strictly positive when $\sigma > 0$. □

Applying the approximation method of Section 2.4 and by Proposition 2.2 we see that scheduling with the lowest mean first is optimal when variances are equal and ω is at least a half. Summarizing the results in Figure 2.5, we see that a return client should precede any client with a higher mean and standard deviation. In addition, the diagonal where CV is fixed delineates two regions. Above the diagonal the expected idle and waiting times increase, whereas in the cone below the diagonal it is known only that the objective function will increase as long as $\omega \geq 0.5$.

Example 2.2. *Typically in healthcare, the value of ω is close to* 1, *see Robinson and Chen (2011). So, even in the case that the variances of both new and return clients of Example 2.1 were alike, it is still sequentially optimal to schedule return clients first.*

2.7 Numerical Assessments

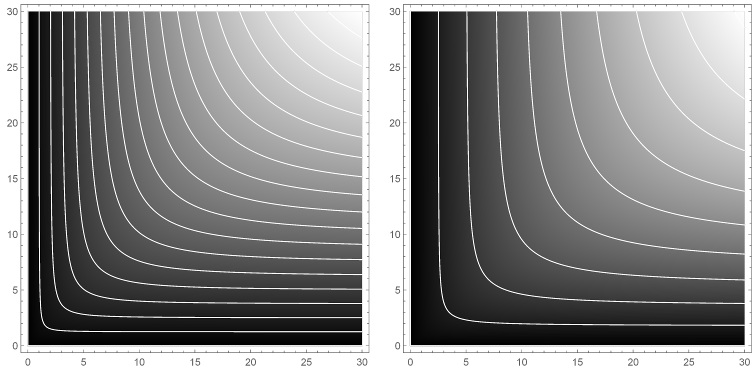
In the previous section, we proved that increasing mean or increasing the standard deviation increases the sequentially minimized objective function. The cases in which one of them increases and the other decreases are intractable for any mathematical formulation. For these instances we rely on numerical computations of the behavior of the minimized objective function.

By the approximation of Section 2.4 and Proposition 2.2 it suffices to look at one client at a time by recomputing the moments, so we provide the following contour plots for $\omega = 0.5$ and 0.8 in Figure 2.6. The contours in the plot depict levels on which the minimized objective function has the same value. In addition, the lighter the area is, the higher the value of the minimized objective function.

In the case of Figure 2.6a, the isocost curves of the minimized objective function are characterized by

$$\mathbb{SD}[B] = \mathbb{E}[B]\sqrt{\mathrm{e}^{\Phi^{-1}\left(\frac{2C+\mathbb{E}[B]}{\mathbb{E}[B]}\right)^2} - 1},$$

in which C is the value of the minimized objective function, see Eq. (2.28). These isocost curves can be used to address the sequencing question further than just increases in the mean or standard deviation. In practice, due to the speed of fitting moments it would also suffice to compare the losses for all as-yet unscheduled clients and to schedule the most advantageous client next. The contour plots in this case serve to aid visualization of the relationships between different prospective clients. By iteratively estimating the moments via the moment-iteration method, one can determine which client to schedule next as to minimize the objective function.



(a) $\omega = 0.5$.

(b) $\omega = 0.8$.

Figure 2.6: Isocost curves of the minimized objective function with along the horizontal the mean and along the vertical axis the standard deviation. The cost difference between each curve equals 0.5 units.

In Theorem 2.2 we showed that the cost is always increasing in the mean when $\omega \geq 0.5$, a reasonable presumption in appointment scheduling. Table 2.4 compares losses for $\mathbb{E}[B] = 1$ and $\mathbb{E}[B]' = \mathbb{E}[B] + 0.01$ while holding variance constant, and depicts the sign of the change. This table shows examples of the inverse, where cost *decreases* while mean increases, which is important behavior to note if dealing with instances where it is reasonable to anticipate an $\omega < 0.5$. This effect is most pronounced when the coefficient of variation is small. It suffices to construct this table for only one value of $\mathbb{E}[B] = e^{\mu+\sigma^2/2}$ and for different values of the coefficient of variation, $\text{cv} = \sqrt{e^{\sigma^2} - 1}$, as we have shown in Eq. (2.29) that the derivative of the optimal cost in the mean does not depend upon the parameter μ.

CV	ω					
	0.01	0.1	0.3	0.4	0.45	0.5
$1/20$	−	−	−	−	−	+
$1/10$	−	−	−	−	+	+
$1/5$	−	−	−	+	+	+
$1/2$	−	−	+	+	+	+
1	−	+	+	+	+	+

Table 2.4: The effect of increasing the mean for various combinations of CV and $\omega \leq 0.5$.

Tables 2.5 and 2.6 compare simultaneous and sequential schedules. To make the two schedules comparable we make the choice to fix the makespans of the two schedules, represented via the last client's completion time: $\mathbb{E}[C_{10}]$. This enables us to capture the effect of including overtime in the objective function $\mathscr{F}(\boldsymbol{x}; \omega, \gamma)$, supporting the proposition in Section 2.3.3 that one can control overtime implicitly through the parameter ω.

In Table 2.5 we consider a schedule without regard for overtime. $\omega = 0.8$ has been chosen for the simultaneous schedule as it is considered a reasonable reflection of practice. As in Figures 2.3 and 2.4, we also consider the cost ratios which are defined for client i by $\rho_i :- \mathbb{E}[I_i]/\mathbb{E}[W_i]$. In Table 2.6 we consider a schedule with a weight on overtime of $\gamma = 1.5\omega$. This leads to an anticipated reduction in expected overtime. It is interesting to note that simply by matching makespans we are able to reduce expected overtime for the sequential schedule as well, and that weighting overtime in the simultaneous schedule brings its solution closer to that of the sequential schedule.

Previously we considered sequential optimality, which involved generating a sequence by appending the client who would add the lowest additional cost at each iteration. To do so we derived sequencing rules, but these rules are not yet a guarantee that under the sequential optimization paradigm this sequence yields the lowest total cost as in Eq. (2.2). So, it leaves the question whether the sequencing rule in our moment-iteration framework is also *overall optimal.*

For the case of log-normal service times the optimality of the `SCF` rule cannot be established, rather we focus on a typical healthcare setting (Example

	Simultaneous schedule with $\omega = 0.8$				Sequential schedule with $\omega = 0.7047$			
i	$\bar{x}_i$	$\mathbb{E}[I_i]$	$\mathbb{E}[W_i]$	ρ_i	x_i^*	$\mathbb{E}[I_i]$	$\mathbb{E}[W_i]$	ρ_i
1	0.6548	0	0	-	0.6364	0	0	-
2	0.9622	0.0562	0.4014	0.1401	0.9174	0.0506	0.4143	0.1222
3	1.0179	0.0882	0.5273	0.1672	0.9856	0.0712	0.5681	0.1253
4	1.0331	0.0982	0.6077	0.1616	1.0150	0.0796	0.6620	0.1202
5	1.0360	0.1006	0.6752	0.1490	1.0326	0.0848	0.7318	0.1158
6	1.0325	0.0990	0.7382	0.1341	1.0448	0.0885	0.7878	0.1124
7	1.0236	0.0948	0.8005	0.1184	1.0539	0.0914	0.8344	0.1096
8	0.9991	0.0885	0.8653	0.1022	1.0611	0.0938	0.8742	0.1073
9	0.9037	0.0774	0.9437	0.0821	1.0669	0.0957	0.9089	0.1053
10	-	0.0500	1.0900	0.0459	-	0.0974	0.9394	0.1037
Session	$\mathbb{E}[C_{10}] = 10.7530$		$\mathbb{E}[O] = 0.9048$		$\mathbb{E}[C_{10}] = 10.7530$		$\mathbb{E}[O] = 0.8888$	

Table 2.5: Comparing a simultaneous to a sequential schedule by setting makespan equal. $\mathbb{E}[B_i] = 1$, CV $= 0.6$, for all i. Planned session due date, $\mathrm{T} = \sum_i \mathbb{E}[B_i] = 10$. No overtime included in the objective function, as in Eq. (2.1).

2.1) to make a numerical assessment of all possible permutations under the sequential optimization procedure to conclude that our sequencing result likely holds. However, for the case of exponential service times, the SCF rule is indeed overall optimal as we will show at the end of the next section.

Log-Normal

Unfortunately, even using the tractable moment-iteration method, the partial objectives in case of the log-normal distribution do not nicely add up. So, to get a better grip on the overall sequencing question here, we compare all schedules in a case study. Using the algorithm outlined in Section 2.4 to schedule 3 new (N) and 6 return (R) clients with the same characteristics as in the example. Furthermore we choose $\omega = 0.8$ and do not include overtime, $\gamma = 0$. In Figure 2.7, for all $\binom{9}{3} = 84$ possible sequences the total cost has been computed for sequentially optimized schedules, as to complete the comparison also the schedules obtained by simultaneous optimization (Eq. (2.1)) are provided, which depict the same behavior.

In Example 2.1 we learned that a return should precede a new client to minimize the sequential objective function, $\mathscr{F}_i(x_i^*; \omega)$. Comparing the simultaneous objective function with sequential schedule $\mathscr{F}(\boldsymbol{x}^*; \omega)$ we see that each time a new client is put further towards the end of the schedule, the objective function decreases. This explains the characteristic 'sawtooth' behavior in the figure; starting from NNNRRRRRR and finally arriving at the sequence of RRRRRRNNN, resulting in a 14% reduction in the objective function.

Exponential

In case of exponential service times, the clients' sojourn-time distributions do not reduce to exponential distributions, but to phase-types (Wang 1999, Kuiper

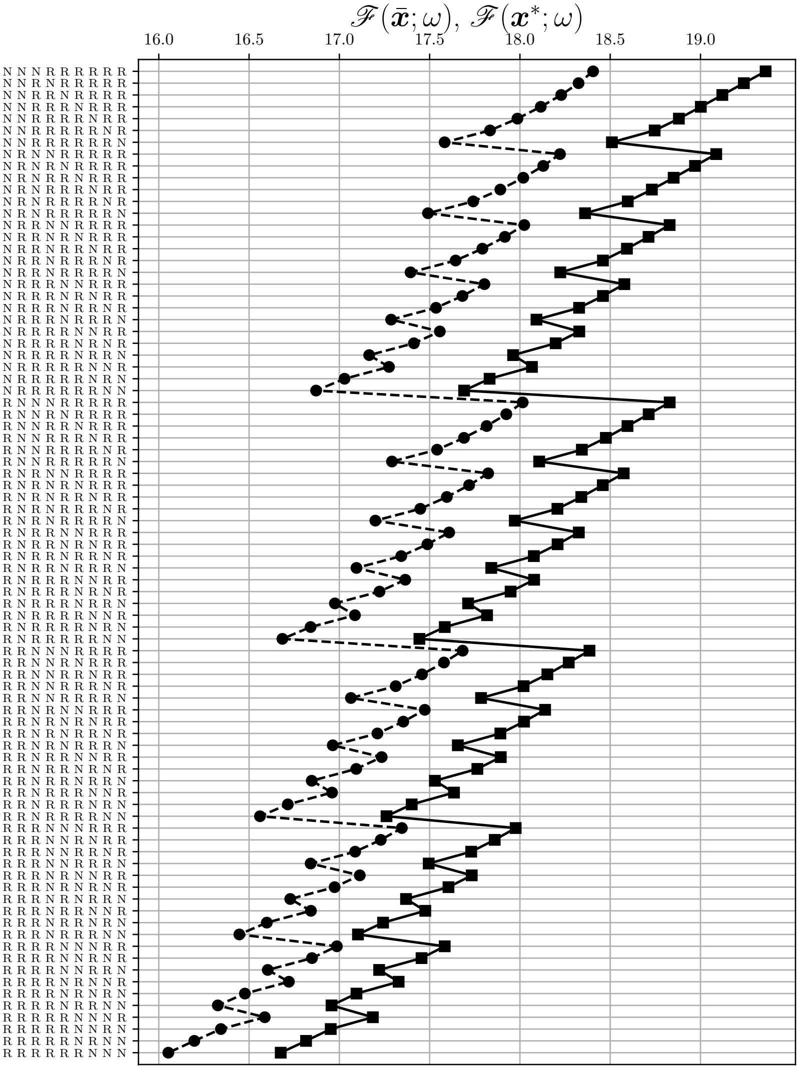


Figure 2.7: Aggregate, i.e. simultaneous, costs of all possible sequential schedules $\boldsymbol{x}^*$ (solid) with 3 new and 6 return clients, with client characteristics in line with Example 2.1. For comparison, also the schedules $\bar{\boldsymbol{x}}$ (dashed) from simultaneous optimization are added.

	Simultaneous schedule with $\omega = 0.8$				Sequential schedule with $\omega = 0.7854$			
i	$\bar{x}_i$	$\mathbb{E}[I_i]$	$\mathbb{E}[W_i]$	ρ_i	x_i^*	$\mathbb{E}[I_i]$	$\mathbb{E}[W_i]$	ρ_i
1	0.5433	0	0	-	0.5531	0	0	-
2	0.8725	0.0273	0.4840	0.0565	0.8510	0.0294	0.4763	0.0617
3	0.9472	0.0467	0.6583	0.0710	0.9301	0.0432	0.6685	0.0647
4	0.9722	0.0544	0.7655	0.0711	0.9629	0.0490	0.7874	0.0623
5	0.9834	0.0577	0.8510	0.0678	0.9821	0.0526	0.8770	0.0599
6	0.9886	0.0588	0.9264	0.0635	0.9951	0.0551	0.9501	0.0580
7	0.9904	0.0585	0.9962	0.0587	1.0048	0.0572	1.0121	0.0565
8	0.9890	0.0572	1.0630	0.0538	1.0124	0.0588	1.0661	0.0552
9	0.9805	0.0550	1.1290	0.0487	1.0185	0.0602	1.1139	0.0540
10	-	0.0513	1.1998	0.0427	-	0.0614	1.1567	0.0531
Session	$\mathbb{E}[C_{10}] = 10.4669$		$\mathbb{E}[O] = 0.7699$		$\mathbb{E}[C_{10}] = 10.4669$		$\mathbb{E}[O] = 0.7657$	

Table 2.6: Comparing a simultaneous to a sequential schedule by setting makespan equal. $\mathbb{E}[B_i] = 1$, CV $= 0.6$, for all i. Planned session due date, $\mathrm{T} = \sum_i \mathbb{E}[B_i] = 10$. Overtime included in the objective function with weight $\gamma = 1.5\omega$ as in Eq. (3.3).

et al. 2015). However, by using the moment-iteration method we do end up with an exponential distribution each time, which we can exploit in Theorem 2.3. This result augments on the work by Choi and Wilhelm (2020), in which sequencing results for three or more clients are obtained in case of equidistant schedules.

Theorem 2.3. *Using sequential scheduling in combination with the moment-iteration method for exponential service times, sequencing lowest mean/variance first is also overall optimal.*

Proof. First note that an expression for the ith client's finish time is:

$$C_i = \sum_{j=1}^{i-1} x_j + S_i = \sum_{i=1}^{j} (B_i + I_i),$$

so that subtracting two subsequent finish times, i.e., $C_i - C_{i-1}$, leads to the following equation

$$I_i = x_{i-1} + W_i - S_{i-1},$$

given that we are under the sequential optimization paradigm we evaluate the sequential solution of Eq. (2.21) to get (cf. Eq. (2.24))

$$\mathscr{F}_i(x_i^*; \omega) = \omega x_i^* + \mathbb{E}W_{i+1}(x_i^*) - \omega\ \mathbb{E}S_i = -\frac{\omega \log(\omega)}{\mu_i}. \tag{2.31}$$

At the same time because of the moment-iteration method, the following relation can be deduced

$$\frac{1}{\mu_i} = \sum_{j=1}^{i} \frac{\omega^{i-j}}{\lambda_j}, \tag{2.32}$$

which can be plugged into the sum of the partial objectives displayed in Eq. (2.31) to reformulate the overall objective as

$$\begin{aligned}\mathscr{F}(\boldsymbol{x}^*;\omega) &= -\omega\log\omega\sum_{i=1}^{n}\sum_{j=1}^{i}\frac{\omega^{i-j}}{\lambda_j} = -\omega\log\omega\sum_{j=1}^{n}\frac{1}{\lambda_j}\sum_{i=j}^{n}\omega^{i-j}\\ &= -\frac{\omega\log\omega}{1-\omega}\sum_{j=1}^{n}\frac{1-\omega^{n+1-j}}{\lambda_j}.\end{aligned}$$

Because $\omega \in (0,1)$, the term $1-\omega^{n+1-j}$ becomes smaller when $j \to n$. So, to minimize the objective the client with the greatest λ_j should precede, which constitutes scheduling clients with smallest mean/variance first. □

2.8 Robust Sequential Optimization

In Mak et al. (2015) robust optimization is employed to deal with incomplete information about the service-time distribution. To that end we propose an approach that straightforwardly fits in the sequential optimization framework. Another relevant question is how to obtain schedules that are robust against mis-specification of the weight parameter ω of Eq. (2.1). This value has to be set by the practitioner and we will show that there exists a value to guarantee a worst-case performance, which can be used as a starting point when relying on sequential optimization.

2.8.1 Sequential Optimization with Limited Distributional Information

In Gallego and Moon (1993) a robust variant of the newsvendor problem is introduced which minimizes the worst-case distribution. It is also shown that this distribution, a two-point distribution, is unique. Translating that result to our sequential optimization framework and combining it with the moment-iteration method, we can apply it to our case, which iteratively yields the sequentially optimal scheduling and sequencing decision.

As established, the optimal inter-arrival time that minimizes the worst-case distribution is

$$x^* = \mathbb{E}[B] + \frac{\sqrt{\mathbb{V}ar[B]}}{2}\left(\sqrt{\frac{1-\omega}{\omega}} - \sqrt{\frac{\omega}{1-\omega}}\right). \tag{2.33}$$

Using this optimal inter-arrival that minimizes the worst-case distribution, the worst-case distribution for a variable B becomes:

$$f_B(t) = \begin{cases}\mathbb{E}[B] - \sqrt{\mathbb{V}ar[B]\frac{\omega}{1-\omega}} & \text{if } t = 1-\omega,\\ \mathbb{E}[B] + \sqrt{\mathbb{V}ar[B]\frac{1-\omega}{\omega}} & \text{if } t = \omega.\end{cases} \tag{2.34}$$

Employing these expressions for the sojourn-time distribution at each iteration, we replace B by S_{i-1}, we arrive at the following update scheme for the moment-iteration method:

$$\mathbb{E}[S_i] = \mathbb{E}[B_i] + \sqrt{\mathbb{V}ar[S_{i-1}]}\frac{\sqrt{\omega}}{2\sqrt{1-\omega}}; \tag{2.35}$$

$$\mathbb{V}ar[S_i] = \mathbb{V}ar[B_i] + \mathbb{V}ar[S_{i-1}]\frac{\sqrt{\omega}\sqrt{1-\omega}}{4}. \tag{2.36}$$

Theorem 2.4. *Using the worst-case distribution with the moment-iteration method, smallest variance clients should be sequentially scheduled first.*

Proof. The expected idle and waiting times are

$$\mathbb{E}[I_i] = \sqrt{\mathbb{V}ar[S_{i-1}]}\frac{\sqrt{1-\omega}}{2\sqrt{\omega}} \quad \text{and} \quad \mathbb{E}[W_i] = \sqrt{\mathbb{V}ar[S_{i-1}]}\frac{\sqrt{\omega}}{2\sqrt{1-\omega}},$$

so that the total cost at each iterand becomes:

$$\mathscr{F}_i(x_i^*;\omega) = \sqrt{\mathbb{V}ar[S_{i-1}]}\frac{\sqrt{\omega}\sqrt{1-\omega}}{2}.$$

So, the costs for client i solely depend on the variance—not the mean—of the sojourn time of its predecessor. Observing Eq. (2.36) we find that (higher) variances are propagated to subsequent clients:

$$\mathscr{F}(\boldsymbol{x}^*;\omega) = \frac{\sqrt{\omega}\sqrt{1-\omega}}{2}\sum_{i=1}^{n}\sqrt{\mathbb{V}ar[B_i] + \sum_{j=1}^{i-1}\left(\frac{\sqrt{\omega}\sqrt{1-\omega}}{4}\right)^{i-j}\mathbb{V}ar[B_j]}.$$

From which we immediately deduce that these costs are minimal when sequenced `SVF`. □

To ensure non-negative arrival times, we afterwards apply the $(\cdot)^+$ operator on the inter-arrival times $\boldsymbol{x}^*$. Note that a negative value only occurs when $\omega > 0.5$ and we deal with a relative high coefficient of variation CV, that is, $\text{cv}_{S_{i-1}} :- \frac{\sqrt{\mathbb{V}ar[S_{i_1}]}}{\mathbb{E}[S_{i-1}]} > \frac{2\sqrt{(1-\omega)\omega}}{2\omega-1}$. Practically, in these cases there is no idling; instead a buffer of waiting clients is built up. Finally, note that the worst-case distribution is a *location-scale* family and that Theorem 2.4 can be generalized to hold for any *location-scale* family. So, under sequential optimization, sequencing `SVF` also applies to the case of, for example, normal and uniform distributions as sometimes used in scheduling (Denton and Gupta 2003).

2.8.2 Guaranteed Worst-Case Performance

To guarantee a worst-case performance, we assess schedules obtained by the sequential optimization approach in terms of the overall objective of minimizing sums of idle and waiting times as in Eq. (2.1) for different weight parameters ω.

Suppose that the true desired weighting parameter of the optimization problem is uncertain. In Figure 2.8, for example, we first generate *sequential* solutions $\boldsymbol{x}^*(\psi)$ (defined by $x_i^*(\psi) = F_{S_i}^{-1}(1-\psi)$) for varied ψ and plot the simultaneous objective function $\mathscr{F}(\boldsymbol{x}^*(\psi); \omega, \gamma)$ for candidate true weights $\omega = \{0.05, 0.5, 0.95\}$, with $\gamma = 1.5\omega$. The resulting costs are the costs of the simultaneous objective function for each value of ω, using the solution $\boldsymbol{x}^*(\psi)$. Note that there appears to be a point where all three of these lines cross, i.e., there is some point ψ^* for which the simultaneous objective functions all return the same value, regardless of ω. This point can be identified as minimizing the regret under misspecification of ω, as at any other point $\psi \neq \psi^*$ there is some ω for which the cost will be higher. ψ^* thus describes:

$$\inf_{\psi\in\Psi} \sup_{\omega\in\Omega} \left(\mathscr{F}(\boldsymbol{x}^*(\psi); \omega, \gamma) - \mathscr{F}^*(\boldsymbol{x}^*(\psi); \gamma)\right) = 0,$$

where $\mathscr{F}^*(\boldsymbol{x}^*(\psi); \gamma) = \inf_{\omega\in\Omega} \mathscr{F}(\boldsymbol{x}^*(\psi); \omega, \gamma)$. The existence of such a point under mild conditions is proven in the next theorem.

Theorem 2.5. *There is a point $\psi^* \in (0,1)$ such that the schedule generated by the sequential approach results in the same aggregate, i.e., simultaneous, cost without overtime, $\mathscr{F}(\boldsymbol{x}^*; \omega)$, for all choices of weight ω.*

Proof. The existence of a sequential schedule which balances expected idle and waiting time can be shown by looking at the fraction $D(\boldsymbol{x}^*(\psi)) = {}^{\sum \mathbb{E}W_i}/_{\sum \mathbb{E}I_i}$, which should equal 1 to balance both costs such that any ω will result in the same cost. Since $\lim_{\psi\uparrow 1} D(\boldsymbol{x}^*(\psi)) = \infty$ and $\lim_{\psi\downarrow 0} D(\boldsymbol{x}^*(\psi)) = 0$, we invoke the intermediate value theorem to guarantee existence of such a ψ^*. Now, since waiting times are monotonically increasing whilst idle times decreasing in ψ we have uniqueness. □

Conditions can also be derived for which the above result extends to the aggregate, i.e., simultaneous, cost including overtime, $\mathscr{F}(\boldsymbol{x}^*; \omega, \gamma)$. Let

$$D_O(\boldsymbol{x}^*(\psi)) = \frac{\sum_{i=1}^n \mathbb{E}[W_i]}{\sum_{i=1}^n \mathbb{E}[I_i] + \alpha\mathbb{E}[O]},$$

(where again $\gamma = \alpha\omega$) this fraction must again be somewhere equal to 1 for the existence of a robust point. Consider a target session-end time of T = 0, as this maximizes overtime. We have again on the one hand that $\lim_{\psi\downarrow 0} D_O(\boldsymbol{x}^*(\psi)) = 0$, but on the other hand $\lim_{\psi\uparrow 1} D_O(\boldsymbol{x}^*(\psi)) = {}^{\sum_{i=1}^n (n-i)\mathbb{E}[B_i]}/_{\gamma\mathbb{E}[O]}$. As long as neither $\mathbb{E}[B_n]$ nor γ are too large this will hold to be greater than one. The resulting sequential schedules constructed with the resulting value ψ^* are robust against choosing a wrong value for ω and thereby offer a guaranteed worst-case performance.

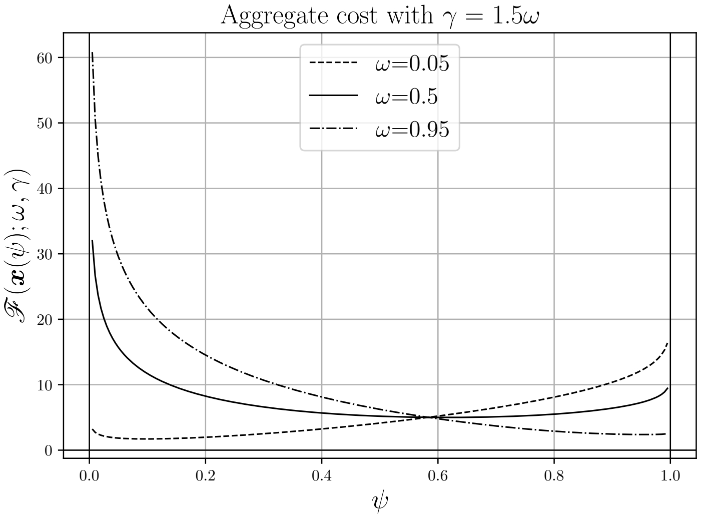


Figure 2.8: Calculating the aggregate, i.e. simultaneous, cost of a sequential schedule taking overtime into account. Each client has a mean of 1 and a CV of 0.6.

2.9 Conclusion and Discussion

We provide an efficient approach to optimize the appointment scheduling problem. First, assuming that service times are distributed according to a distribution, e.g., exponential or log-normal, we apply a moment-iteration method to the appointment scheduling problem to make it computationally feasible. The method is based on iteratively approximating the *sojourn time* by, for example, a log-normal distribution with the same moments, but can be applied to any type of distribution. In this way we provide accurate solutions near instantaneously, and the solutions result in the well-known dome-shape pattern (Denton and Gupta 2003, Kaandorp and Koole 2007, Kuiper et al. 2015). Second, we establish guidelines to help practitioners decide in which order to schedule clients. Such a comprehensive consideration of the problem is quintessential in healthcare where clients visit physicians, or are to be scheduled an MRI, CT-scan or X-ray.

Focusing on the sequential version of the appointment scheduling problem (Wang 1993, Kemper et al. 2014, Kuiper et al. 2015), which entails minimizing the objective function for each client enabling full control over the distribution of costs per client, we consider the setting of heterogeneous clients; a particularly challenging problem in literature. Using the moment-iteration method, we gain insights into who and when to schedule next by considering the mean, variance or both. Zooming in on the practically relevant case of log-normal service times, we show that amplifying variance always increases expected idle and waiting times. Also, when the mean and standard deviation increase at the same rate the expected idle and waiting times rise when considered in the optimum. Finally, when only the mean increases, we prove that when the weight parameter ω is chosen such that idle time is reflected to be more or equally important than waiting time, which is easily met in healthcare, it is sequentially optimal to schedule lower-mean clients first. When mean and standard deviation move in opposing directions we provide contour plots for typical weightings of idle and waiting time.

In addition, our results extend to the setting of surgery scheduling (Guda et al. 2016), as we show that under the sequential scheduling paradigm there is a straightforward connection between the problems of finding appointment times and due date determination. It uncovers an attractive property of sequential optimization, namely that the due date, i.e., the expected finish time of a client, will be the moment that the next client arrives.

As the approach depends on the choice of distribution to approximate the sojourn time distribution and a weight parameter in the cost function, we show how to utilize the approach when these elements are unknown. First, when no distributional information is available, but only the mean and variance are known, we amend the approach with a distribution-free model by incorporating a worst-case distribution (Gallego and Moon 1993). Next, we demonstrate the existence of sequential solutions that are robust against misspecification of the weighting parameter.

One of the limitations of the study is that it focuses on a sequential optimization setting of the problem, in which by means of the moment-iteration method interesting sequencing results are obtained. A natural question is to investigate whether in different optimization frameworks or by using other scheduling heuristics similar results, such as sequencing clients smallest-mean/variance first, can be obtained. It is however known that in the traditional optimization framework of appointment scheduling, in which the objective function is jointly minimized over all clients' arrivals (see Eq. (2.1)), the answer is negative (Jafarnia-Jahromi and Jain 2020, Kong et al. 2016).

A direction for future research is to expand to a more integrated rendering of the problem that outpatient healthcare clinics face by including capacity constraints and client flow within a clinic. For example, along the lines of the research by Gul et al. (2011) and White et al. (2011), who use discrete-event simulation to find out how various factors and heuristics interact. Their experiments reaffirm that, in practice, it is wise to schedule low mean and low variation clients first. Also, one can consider a multi-server setting, as the one studied in Kuiper and Lee (2022). Lastly, the consideration of different performance metrics is an avenue for further research; for example, to consider a quantile objective (Sang et al. 2021) or incorporate quadratic idling or waiting costs as explored in Kuiper et al. (2023).

Chapter 3

On Scheduling Multiple Servers

3.1 Introduction

Appointment schedules are often used in settings where resources are scarce; as a consequence appointment schedules are used in healthcare. Current literature mainly focuses on the single-server setting. In healthcare such a setting is often appropriate as continuity of care is obeyed: patients see the same physician during the course of their treatments.

Some settings, however, do not fit well into this framework, such as magnetic resonance imaging (MRI), X-ray facilities and operating rooms. Each of these cases has multiple resources that are present and it is logical that the next patient will be served by the first resource to become available. Other examples can be found in (e.g., El-Sharo et al. 2015, Soltani et al. 2019), such as legal counselling, technical support appointments, visa application processes, dental hygiene services and medical rehabilitation services. Especially in healthcare, our analysis will demonstrate the benefits when relaxing the continuity of care restraint in situations where multiple service providers are able to provide the same service. In fact Green et al. (2013) conclude that, to fulfil the growing need of primary care in the near future, pooling of physicians is inevitable.

The benefits of increased flexibility by pooling resources are assumed reduction of patient or client waiting times and increased utilization. The decision whether to pool resources, such as MRIs or physicians, is of a *tactical* nature, as it concerns the assignment of clients to resources, see Hulshof et al. (2012) and Ahmadi-Javid et al. (2017).

A complicating factor in appointment scheduling is random service times (Ho and Lau 1992, Çayırlı and Veral 2003). This randomness is reflected in a combination of:

- Idling resources due to having excessive capacity, resulting in *idle time.*
- Session overruns due to insufficient capacity towards the end of the schedule, creating *overtime.*
- Waiting clients as a result of insufficient capacity, resulting in *waiting time.*

An appointment schedule tries to find a balance by minimizing a weighted sum of these ramifications of under- and over-capacity.

To define the problem in a mathematical framework, let there be s servers and $n > s$ clients to be scheduled. Let t_i and B_i be the arrival and service time of the i-th client. Assume that each server starts by serving one of the initial s clients, so $t_1 = t_2 = \ldots = t_s = 0$, so that for each subsequent client $i \in \{s+1\ldots,n\}$, W_i denotes the waiting time, and I_i the idle time, which is the time that resources are idle before the i-th client's arrival. In addition, define the overtime O as the time that the session runs over the scheduled session-end time T. The objective is to determine a schedule, given by arrival epochs $(t_{s+1},\ldots,t_n)$ of the $n-s$ subsequent clients as to minimize the total expected idle time, waiting time and overtime over the session. By considering the inter-arrival time between two appointment epochs $x_{s+i} := t_{s+i} - t_{s+i-1}$, the problem can equivalently be formulated as:

$$\min_{(x_{s+1},\ldots,x_n)} \sum_{i=s+1}^{n} (c_I \mathbb{E} I_i + c_W \mathbb{E} W_i) + c_O \mathbb{E} O, \tag{3.1}$$

wherein c_I, c_W and c_O are weight parameters to be chosen at the discretion of the practitioner.

The computation of these metrics is typically done by considering the evolution of the schedule as a queue. In our case the resulting queueing system is, using Kendall's notation, a D/G/s queue: deterministic inter-arrival times (not necessarily uniformly spaced), general service times, and s servers.

Our methodology relies on the fact that service times can be approximated well by mixtures of exponentials, i.e., phase-type distributions, wherein each exponential distribution can be thought of as a state of the system. Such a description can be extended to the multi-server setting for which we derive a tractable recursive procedure by exploiting its semi-Markovian nature to keep track of the system, allowing evaluation of the objective function. For optimized multi-server appointment schedules we find that the resulting inter-arrival times feature singular patterns that do not appear in the single-server setting. Further, comparing equivalent configurations in number of servers, the potential gains of pooling in appointment scheduling with random service times are quantified, as well as the impact of various environmental and free parameters, such as randomness of the service times, the impact of the weight parameters, and the occurrence of no-shows.

3.2 Literature Review

We divide literature on appointment scheduling into two streams: single-server systems and multi-server systems. The former class has been studied extensively; we refer for comprehensive reviews on these efforts to Çayırlı and Veral (2003), Gupta and Denton (2008), Ahmadi-Javid et al. (2017). Study on multi-server systems has usually been restricted to the domain of multi-stage settings. Few works have focused on the single-stage, multi-server setting for appointment scheduling, i.e., the $D/G/s$ queue which is analytically explored in this paper. Below we highlight work that relates to our research.

3.2.1 Single-Server, Single-Stage Environments

The single server setting, that is $s = 1$, is naturally applicable to the single stage case. This does not, however, constrain the framework from being applicable to a multi-stage setting, for example when other stages have more than sufficient capacity not to be a bottleneck, e.g., a reception. In Welch and Bailey (1952) the single-server environment was first formulated. These authors also formulated the well-known Bailey-Welch appointment rule, which assigns multiple clients to the first slot to circumvent possible idle time in early stages of the schedule. Ho and Lau (1992) study variations on this appointment rule and find that among important environmental factors affecting the performance of an appointment schedule the most important are the number of clients to be scheduled, service-time variability and no-shows.

Another stream of research aside from the study of appointment rules is that of developing methods for finding optimal arrival epochs. An example is the work by Denton and Gupta (2003), who introduce a sequential bounding approach in which the problem is framed as a linear program. Using the L-shaped algorithm, they successively partition the outcome space to approximate an optimal solution. Klassen and Yoogalingam (2009) use simulation in conjunction with optimization to address the single-stage appointment scheduling problem. Another paradigm is to solve this problem over a discrete grid, such as in Kaandorp and Koole (2007), who assume exponential service times. Zacharias and Yunes (2020) show the concept of multi-modularity to hold for general stochastic service times, which guarantees the success of efficient optimization algorithms.

A common method to obtain tractability is the use of phase-type distributions (Asmussen et al. 1996), which have proven to provide good levels of accuracy. In the context of appointment scheduling, Wang (1997) is the first work which employs phase-type distributions to derive a recursive system. In the same stream, Bosch and Dietz (2001) use phase-type distributions to analyze the waiting time and overtime over a grid of schedules and show submodularity to assure convergence. Kuiper et al. (2015) introduce a general method to approximate service times by a phase-type counterpart, allowing computation of relevant queue metrics and facilitating steady-state analyses. They show that it provides good approximations for both the log-normal and Weibull distributions.

Another approach is to discretize time, such an approach is followed by De Vuyst et al. (2014) and Begen and Queyranne (2011) to facilitate evaluation and optimization. Lastly we name the work by Mak et al. (2015), who study appointment scheduling considering worst-case distributions, which we show to be closely related to our results obtained in steady state.

Focusing on the solutions that these approaches produce, many have reported that the optimized inter-arrival times depict a *dome-shape pattern* (Wang 1997, Denton and Gupta 2003, Kaandorp and Koole 2007, Hassin and Mendel 2008, Klassen and Yoogalingam 2009, Kuiper et al. 2015). Appointments early in the session and towards the end are more condensed, whereas in the middle the inter-arrival times between appointments are lengthier.

3.2.2 Multi-Stage Environments

As noted in Çayırlı and Veral (2003) and Ahmadi-Javid et al. (2017) the majority of the literature focuses on single-server appointment scheduling. However, multi-server settings are nevertheless prevalent in healthcare. The first extension to consider is the addition of servers in series, creating a multi-stage environment. For example, a client may first have an X-ray and then have an appointment with a specialist.

Rising et al. (1973) study a system of multiple stages at a university outpatient clinic by means of Monte Carlo simulation. Cox et al. (1985) develop and simulate a queueing model for the multi-stage setting found in an ear, nose & throat outpatient clinic. Also relying on simulation, White et al. (2011) study a system in which – besides introducing capacity constraints – they distinguish between two patient types, one of which requires an X-ray before appointment. In surgery scheduling, Saremi et al. (2013) use simulation optimization in order to address a multi-stage operating room scheduling problem, incorporating the availability of surgeons.

Another sequential service setting is studied in Zhou and Yue (2019), in which they introduce a stochastic linear program, which is solved by combining a sample average approximation and linear programming (cf. Denton and Gupta 2003). A two-stage, tandem setting is studied analytically in Kuiper and Mandjes (2015). Klassen and Yoogalingam (2019) study by means of simulation the clinic's effectiveness when part of the physician's work is taken over in an earlier stage by assistants.

Finally, we mention research that considers systems with multiple stages and servers. Most of this work is case specific and relies on simulation, e.g., Côté and Stein (2007). Another example is the work by Alvarez-Oh et al. (2018), who study a simple system in which patients have to be seen by one of two nurses and then a dedicated provider (single server). Mandelbaum et al. (2020) develop a data-driven robust optimization approach based on uncertainty sets that accommodates a multi-server setting with various patient flows. They apply their model on a cancer center's infusion units and report a 15% to 40% reduction of waiting and overtime costs.

3.2.3 Multiple Parallel Servers

Another extension of the single-server framework is that of servers in parallel. In the specific setting where there is server preference we refer to (Section 5.1 Ahmadi-Javid et al. 2017) and references therein. Here we focus on the case where clients are indifferent to servers.

Denton et al. (2010b) study the problem of scheduling surgeries to multiple operating rooms (parallel servers), where in a first stage it is decided how many servers to open, such that in the second stage surgeries are assigned to specific operating rooms. Once assigned, each operating room acts as a single-server system. El-Sharo et al. (2015) consider a model to decide how many clients should be overbooked to slots in a multi-server setting. For each server a separate appointment schedule is made. Only when a patient becomes an overflow patient, or a patient is failed to be served in his or her initially assigned slot, will that patient be allocated to any other server.

As the two examples above still make a schedule assigned to a specific resource we find a closer resemblance to our setting in Swisher et al. (2001). They apply discrete event simulation to a clinic to study its performance in a steady state, in this setting the patient does not go to a specific physician. Furthermore Harper and Gamlin (2003) study by means of simulation the impact of various appointment rules. Sickinger and Kolisch (2009) study an appointment schedule with two computer tomography (CT) scanners. These scanners serve the same queue which consists of three patient classes, namely outpatients, for whom the schedule is built, and inpatient and emergency patients, who provide the randomness to be tackled by the design of an appointment schedule. They find that a generalized Bailey-Welch rule performs well.

Zacharias and Pinedo (2017) offset no-shows by providing a recursive method to compute various performance metrics which are optimized by a local search algorithm. Soltani et al. (2019) focus on a legal counselling center where service times are random and model this randomness by matching the first two moments by a discrete service-time distribution; assigning probabilities to multiples of the slot size. More importantly, as optimization turns out to be computationally intensive, a load-based appointment scheduling heuristic is proposed, which provides a performance increase of 16%.

3.2.4 Contribution and Organization

Our approach augments the current literature on multi-server appointment scheduling by providing a computational approach in continuous time that incorporates service-time variability and the occurrence of no-shows, which are considered the major sources of variation that affect the performance of an appointment schedule (Ho and Lau 1992, Hassin and Mendel 2008).

Relying on phase-type approximations, we extend the phase-type recursion introduced for the single server setting (Wang 1997) to the multi-server setting by compressing the state space. For performance measures of interest, such as idle and waiting time, we obtain semi-analytical derivations. After optimization, the inter-arrival times exhibit some striking patterns at the beginning and end of the session that deviate from the dome-shape pattern reported in literature. Further, the approach enables us to quantify the benefits of pooling in appointment scheduling, which addresses the tactical decision on how to allocate resources; an unchallenged question in the literature on appointment scheduling in healthcare (Ahmadi-Javid et al. 2017).

We extend the work to steady state, which enables the evaluation of the performance gain for large numbers of clients and servers effectively. Interestingly, since appointment schedules are often employed in high-utilization environments, we find that the problem cast in the heavy-traffic regime captures the steady state accurately. This insight underpins that the performance improves by a factor of $\sqrt{s}$ when pooling s servers.

The structure of the paper is as follows. In the subsequent Section 3.3, we state the general scheduling problem for the multi-server setting and show the intrinsic complexity of the problem compared to the single-server setting, and we demonstrate our approach to make the problem tractable. In Section 3.4, we explain in detail how the phase-type distribution of a system with s parallel servers can be obtained and how it facilitates computation of key performance metrics. In Section 3.5, we use the methodology to compute optimal schedules for a given number of clients under various settings, so that we can study the form of the optimal solution as well as the gain from combining servers. Then, in Section 3.6, we extend our phase-type methodology to steady state and consider a corresponding heavy-traffic analysis, which enables us to gain a better insight in the benefits of pooling. Finally, we conclude in Section 3.7.

3.3 Problem Definition

Empirical research reports that patients arrive early more often than late and therefore it is typical to assume that patients are punctual (Cox et al. 1985, Çayırlı and Veral 2003). In the interest of generality we will henceforth refer consistently to *clients* and *servers*. Further, we restrict our model to identical servers and homogeneous clients except for a short discussion on the ramifications of relaxing these assumptions. We assume that servers start non-empty and that there are n clients to be scheduled. Furthermore, in line with appointment scheduling literature, clients are served according to a first-come first served discipline, and there is no pre-emption nor sharing of servers.

3.3.1 Single-Server Performance Metrics

In the single-server setting, assuming punctuality we start a session with the first client to arrive at time $t_1 = 0$. Define the inter-arrival time between the i-th client and his predecessor as $x_i := t_i - t_{i-1}$ for $i = 2, \dots, n$. Furthermore, let $x_1 = 0$. Then it is standard by the *Lindley* recursion (Lindley 1952) that the waiting times are defined recursively by the following equation:

$$W_i = \max\{B_{i-1} + W_{i-1} - x_{i-1}, 0\} = \max\{S_{i-1} - x_{i-1}, 0\}, \tag{3.2}$$

where the sojourn time $S_i = W_i + B_i$. Obviously $S_1 = B_1$ as the first client does not have to wait. The session end time, also known as the *makespan*, is defined as the moment when the final client leaves the system, which is at

$t_n + S_n$, which is equal to the sums of idle and service times, i.e., $\sum_{i=1}^{n}(B_i + I_i)$. So that the sum of idle times and overtime are deduced from:

$$\sum_{i=1}^{n} I_i = t_n + S_n - \sum_{i=1}^{n} B_i \quad \text{and} \quad O = \max\{t_n + S_n - \mathrm{T}, 0\}, \tag{3.3}$$

wherein T is the pre-defined targeted session-end time. There have been many methods proposed to compute these metrics (Ahmadi-Javid et al. 2017). Once a method is found, these metrics can be used to evaluate the objective function in display (3.1).

3.3.2 Complexity of a Multi-Server System

Unfortunately in a multi-server setting, despite the fact that the waiting queue is shared these recursions do *not* apply, as noted in the seminal work by Kiefer and Wolfowitz (1955). One of the critical issues is that the i-th departure is not necessarily by the i-th client. Since clients are served by several servers in parallel there can be overtaking; a client is still in service whilst another server can become available to serve the subsequent client, so that eventually the subsequent client can leave the system earlier than his predecessor. Thus the waiting time of one client does not exclusively depend on the sojourn time of its predecessor, which makes the problem considerably more challenging and the Lindley recursion inapplicable.

As phase-type distributions have a state-space representation, they permit keeping track of the *system*. In detail, it is possible to keep track of which client is being served by which server, for this purpose we need to consider tuples of s dimensions, where 0 denotes a server as empty. The domain of each element in the tuple is in the simplest case just the number of possible clients. For example, in Table 3.1 we show the number of configurations when there are two servers and 5 clients to be served. The shaded, empty cells cannot be reached as they contain cases where both servers are serving the same client (black) or that a client jumps from one server to another (gray). A simple computation of the number of possible client configurations reveals that this number increases by $n^2 - n + 2$, where n is the number of clients. In general, in a similar way it can be shown that for s servers the number of possible configurations is $\mathcal{O}(n^s)$.

3.3.3 Compressing the State Space

In line with literature, we assume homogeneous servers and clients. First, without loss of generality we normalize the mean service time to one for all clients, that is, $\mathbb{E}B_i = \mu_i = 1$. Second, we express the service-time variability in terms of the squared coefficient of variation:

$$\textsc{scv} = \frac{\mathbb{V}\mathrm{ar}B_i}{(\mathbb{E}B_i)^2} = \mathbb{V}\mathrm{ar}B_i. \tag{3.4}$$

$(0,0)$		$(0,2)$	$(0,3)$	$(0,4)$	$(0,5)$
$(1,0)$		$(1,2)^*$	$(1,3)$	$(1,4)$	$(1,5)$
$(3,0)$		$(3,2)$		$(3,4)$	$(3,5)$
$(4,0)$		$(4,2)$	$(4,3)$		$(4,5)$
$(5,0)$		$(5,2)$	$(5,3)$	$(5,4)$	

Table 3.1: The 22 possible configurations of five clients on two servers, the starting state is indicated by the asterisk; client 1 on the first server and client 2 on the second server.

As clients are served according to a fist-come first-served discipline, it suffices for the i-th client to keep track of the work ahead of him, that is *number of clients* in and the *status* of the system upon and after his arrival; for these purposes define the variables Y_i and $\boldsymbol{Z}_i$ respectively. Since the status of the system is given by the current phase(s) of the client(s) in service, $\boldsymbol{Z}_i$ is often multidimensional. Its dimension depends on the number of clients that are currently in service. Keeping track of these variables allows the computation of waiting times and also session-end time. As an example, client-i's *waiting time* W_i can be inferred from the moment a server comes available for the i-th client, i.e., when there are fewer than s clients in service:

$$W_i = \inf\{t \geq 0 | Y_i(t) \leq s\} = \inf\{t \geq 0 | Y_{i-1}(t + x_i) < s\}. \tag{3.5}$$

Idle time requires more thought. The makespan denotes the session-end time, multiplying this quantity by s gives the servers' total capacity over the course of the session and evidently contains *all* idle times.

Obviously, in a multi-server setting it is possible that there is no need to keep all servers active throughout the entire session. In Section 3.5.2 we discuss the impact of this additional feature and why not to include this in the objective function. Lastly, referring to Eq. (3.3), the notion of session overtime will be carried over to the multi-server case. In the next section we propose a method that enables tracking of the system in continuous time and thus computation of these performance metrics in expectation.

3.4 Methodology

In this section we outline the method that enables computation of the system. We do so by relying on phase-type distributions, which has been a widely accepted method in queueing (Neuts 1981, Tijms 1986, Asmussen et al. 1996), and appointment scheduling in specific (Wang 1997, Bosch and Dietz 2001, Kuiper et al. 2015). To keep track of the multi-server system we exploit the property that a convolution of multiple phase-type distributions can again be described by a phase-type distribution.

A service-time distribution B_i is approximated by a *phase-type counterpart*

$$B_i \sim \mathrm{PH}(\boldsymbol{\alpha}, \boldsymbol{S}), \tag{3.6}$$

with $\boldsymbol{\alpha}$ a row vector describing initial probabilities, and $\boldsymbol{S}$ the transition matrix.

Following the standard approach, a mixture of two *Erlang* distributions is advised in case the service-time distribution has an SCV smaller than 1, that is $E_{k-1,k}(\mu; p)$, requiring k phases. Furthermore, a *hyperexponential* distribution, $H_2(\mu_1, \mu_2; p)$ with balanced means is used in case of an SCV larger than 1, this has $k = 2$ phases. The middle case SCV $= 1$ corresponds to the exponential distribution and has just one phase. Without loss of generality the service times are set to 1, as in the single server case in Kuiper et al. (2015).

3.4.1 Phase-Type Recursion

We are interested in the bivariate process $(Y_{s+i}(t), \boldsymbol{Z}_{s+i}(t))$ for $i = 0, \dots, n-s$ that describes the full evolution of the system, where we have:

- $Y_{s+i}(t) \in 0, 1, \dots, s{+}i$ clients in the system, as we start with s clients in service, and
- $\boldsymbol{Z}_{s+i}(t) = (Z_1(t), \dots, Z_\xi(t))$, where $\xi = \min\{Y_{s+i}(t), s\}$, and for each $\ell = 1, \dots, \xi$, $Z_\ell(t) \in 1, \dots, k$.

The k denotes the number of phases of the phase-type counterpart. Z_ℓ can be seen without loss of generality as the ℓ-th server, because of homogeneous servers. There are at most ξ servers to record, as there are either Y_{s+i} clients to be served or all s servers are active.

Since there is no distinction between which server serves which client, the number of unique combinations of states depends only on the number of servers and possible phases. The maximum number of states required turns out to be $\sum_{i=1}^{n} k^{\min\{i,s\}}$, which is $\mathcal{O}(n)$; a remarkable reduction compared to naively considering all unique routings, cf. $\mathcal{O}(n^s)$ as in Section 3.3.2.

Corresponding to the bivariate process, we define the probabilities of finding j clients in the system, $j \in \{0, \dots, s+i\}$, and the server(s) in phase(s) $m_\ell \in \{1, \dots, k\}$ for $\ell \in \{1, \dots, \xi\}$:

$$p^{(s+i)}_{j,(m_1,\dots,m_\xi)}(t) = \mathbb{P}\left[(Y_{s+i}(t), \boldsymbol{Z}_{s+i}(t)) = (j, (m_1, \dots, m_\xi))\right].$$

Define the row vector that contains all possible phases for j clients in service by

$$\begin{aligned}\boldsymbol{p}^{(s+i)}_j(t) = \Big(&p^{(s+i)}_{j,(k,\dots,k)}(t), \dots, p^{(s+i)}_{j,(k,\dots,1)}(t), \cdots, \\ &p^{(s+i)}_{j,(1,\dots,k)}(t), \dots, p^{(s+i)}_{j,(1,\dots,1)}(t)\Big),\end{aligned} \tag{3.7}$$

which is a vector of size $k^{\min\{j,s\}}$. The quantity $\boldsymbol{p}_j^{(s+i)}(t)\mathbf{1}$ is the probability that j clients remain in the system t time units after client $(s+i)$'s arrival, where $\mathbf{1}$ is a column vector of appropriate size.

In the special case that all service times are exponentially distributed, the workload vector $Y_{s+i}(t)$ is only needed to describe the evolution of the system, because each client's service comprises just a single phase. Consequently, the $\boldsymbol{p}_j^{(s+i)}(t)$ become singletons that describe the probabilities that j clients remain t amount of time after the arrival of the i-th client.

For the general case, the probabilities by Eq. (3.7) describe the evolution of the system. Since each server starts by serving a client and no new clients have yet arrived, the initial probability vector of the system is given by the following concatenation: $\boldsymbol{\alpha}_s = (\boldsymbol{\alpha} \otimes \cdots \otimes \boldsymbol{\alpha}, \mathbf{0}_{\sum_{j=1}^{s-1} k^j})$; the Kronecker product in this vector is applied exactly $(s-1)$ times. The transition matrix is given by

$$\boldsymbol{S}_s = \begin{pmatrix} \boldsymbol{S}^{(s)} & \boldsymbol{U}^{(s)} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{S}^{(s-1)} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \boldsymbol{U}^{(3)} & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{S}^{(2)} & \boldsymbol{U}^{(2)} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \boldsymbol{S}^{(1)} \end{pmatrix}, \tag{3.8}$$

where $\boldsymbol{S}^{(\ell)}$ (diagonal) and $\boldsymbol{U}^{(\ell)}$ (upper diagonal) are defined recursively $(1 < \ell \leq s)$ by

$$\boldsymbol{S}^{(\ell)} = I_{|S|} \otimes \boldsymbol{S}^{(\ell-1)} + \boldsymbol{S} \otimes I_{|\boldsymbol{S}^{(\ell-1)}|}, \tag{3.9}$$

$$\boldsymbol{U}^{(\ell)} = I_{|\boldsymbol{U}^{(\ell-1)}|} \otimes \boldsymbol{U}^{(1)} + \boldsymbol{U}^{(\ell-1)} \otimes I_{|\boldsymbol{U}^{(1)}|}. \tag{3.10}$$

with $\boldsymbol{S}^{(1)} = 1$, $\boldsymbol{U}^{(1)} = -\boldsymbol{S}\mathbf{1}$, $|\boldsymbol{A}|$ being the number of rows in matrix $\boldsymbol{A}$, and $I_{|\cdot|}$ an identity matrix with $|\cdot|$ rows and columns. In fact, $\boldsymbol{U}^{(1)}$ is the traditional phase-type exit vector that corresponds to service completion. The elements of the transition matrix in Eq. (3.8) can be understood as $\boldsymbol{S}^{(\ell)}$ describing the transitions between states in which ℓ servers are busy and $\boldsymbol{U}^{(\ell)}$ the *exit matrix* that defines the transitions to only $\ell-1$ servers being busy, and thus one server becoming idle.

The vector $\boldsymbol{p}^{(s)}(t)$ is fully described by a phase-type distribution $\mathrm{PH}(\boldsymbol{\alpha}_s, \boldsymbol{S}_s)$.

$$\boldsymbol{p}^{(s)}(t) = \left(\boldsymbol{p}_s^{(s)}(t), \boldsymbol{p}_{s-1}^{(s)}(t), \ldots, \boldsymbol{p}_1^{(s)}(t)\right) = \boldsymbol{\alpha}_s \exp\left(\boldsymbol{S}_s t\right), \tag{3.11}$$

For a phase-type representation of the system after the arrival of all other clients a recursive procedure will be proposed. In the same fashion as for the initialisation, we are interested in the vector after the $(s+i)$-th client has entered the system, with $i = 1, \dots, n-s$; using Eq. (3.7) we find:

$$\begin{aligned} \boldsymbol{p}^{(s+i)}(t) = \Big(&\boldsymbol{p}_{s+i}^{(s+i)}(t), \boldsymbol{p}_{s+i-1}^{(s+i)}(t), \dots, \\ &\boldsymbol{p}_{s+1}^{(s+i)}(t), \boldsymbol{p}_{s}^{(s+i)}(t), \boldsymbol{p}_{s-1}^{(s+i)}(t), \dots, \boldsymbol{p}_{1}^{(s+i)}(t)\Big). \end{aligned} \tag{3.12}$$

Furthermore, the probability of being in the absorbing state of an empty system is found by

$$p_0^{(s+i)}(t) = 1 - \boldsymbol{p}^{(s+i)}(t)\mathbf{1}.$$

To find an expression that tracks these probabilities over time we introduce a phase-type distribution $(\boldsymbol{\alpha}_{s+i}, \boldsymbol{S}_{s+i})$ for each client $i \in \{1, \dots, n-s\}$. The transition matrix $\boldsymbol{S}_{s+i}$ is found by extending $\boldsymbol{S}_s$ by i times, adding $\boldsymbol{S}^{(s)}$ along the diagonal. In addition, we also need to describe the flow from one client being finished to the next one being served. For this purpose we place the transition matrix $\boldsymbol{T}^{(s)}$ along the upper diagonal, which will be defined below,

$$\boldsymbol{S}_{s+i} = \left(\begin{array}{ccccc|cc} \boldsymbol{S}^{(s)} & \boldsymbol{T}^{(s)} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{S}^{(s)} & \boldsymbol{T}^{(s)} & \ddots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \ddots & \ddots & \mathbf{0} & & \\ \vdots & \ddots & \ddots & \boldsymbol{S}^{(s)} & \boldsymbol{T}^{(s)} & \mathbf{0} & \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{S}^{(s)} & \boldsymbol{T}^{(s)} & \mathbf{0} \\ \hline \multicolumn{5}{c|}{\mathbf{0}} & \multicolumn{2}{c}{\boldsymbol{S}_s} \end{array}\right), \tag{3.13}$$

$$=: \left(\begin{array}{c|c} \boldsymbol{S}_i^{\text{wait}} & \boldsymbol{T}_s^{(s)} \\ \hline \mathbf{0} & \boldsymbol{S}_s \end{array}\right), \tag{3.14}$$

where $\boldsymbol{T}^{(s)}$ follows from the recursion

$$\boldsymbol{T}^{(\ell)} = I_{|\boldsymbol{T}^{(\ell-1)}|} \otimes \boldsymbol{T}^{(1)} + \boldsymbol{T}^{(\ell-1)} \otimes I_{|\boldsymbol{T}^{(1)}|}, \quad \text{with} \quad \boldsymbol{T}^{(1)} = -\boldsymbol{S}\mathbf{1} \otimes \boldsymbol{\alpha}.$$

Thus each block $\boldsymbol{S}^{(s)}$ added to the diagonal in Eq. (3.13) corresponds to states where one might find an additional client in the waiting queue. E.g., after the $(s+i)$-th client's arrival there can be at most $i-s$ clients waiting. Analogously, the exit matrix $\boldsymbol{T}^{(s)}$ describes transitions from a saturated system with $j > s$ clients in the system to one with precisely $j-1 \geq s$, cf. $\boldsymbol{U}^{(\ell)}$ for $\ell \in 2, \dots, s$, see Eq. (3.10), which correspond each time to the ℓ-th server becoming available.

The initial probability vector $\boldsymbol{\alpha}_{s+i}$, for $i = 1, \dots, n-s$, captures the intrinsic *recursivity* of the approach. Since on arrival of a subsequent client the system can be saturated, all s servers are busy and the client enters the waiting queue, or is accepted by an available server. The corresponding vector $\boldsymbol{\alpha}_{s+i}$ can be derived from $\boldsymbol{p}^{(s+i-1)}(x_{s+i})$ as defined in Eq. (3.11) and reads for client $s+i$ who arrives x_{s+i} time after his predecessor:

$$\boldsymbol{\alpha}_{s+i} = f\left(\boldsymbol{p}^{(s+i-1)}(x_{s+i}), \boldsymbol{\alpha}\right)$$

$$:= \Big(\boldsymbol{p}_{s+i-1}^{(s+i-1)}(x_{s+i}), \dots, \boldsymbol{p}_{s+1}^{(s+i-1)}(x_{s+i}), \boldsymbol{p}_{s}^{(s+i-1)}(x_{s+i}),\\ \boldsymbol{\alpha} \otimes \boldsymbol{p}_{s-1}^{(s+i-1)}(x_{s+i}), \dots, \boldsymbol{\alpha} \otimes \boldsymbol{p}_{1}^{(s+i-1)}(x_{s+i}),\\ \boldsymbol{\alpha} \otimes p_{0}^{(s+i-1)}(x_{s+i})\Big), \tag{3.15}$$

wherein the states in the first line of (3.15) correspond to saturation and in the second line of (3.15) to the start of service for the new client. Furthermore, note that this vector has k^s entries more than its predecessor. Thus after arrival of the $(s{+}i)$-th client the evolution of the system as in Eq. (3.12) is described by PH$(\boldsymbol{\alpha}_{s+i}, \boldsymbol{S}_{s+i})$.

3.4.2 Computation of Performance Metrics

Knowing the phase-type representation of the system after each client's arrival, we can compute his waiting time by considering an embedded phase-type distribution. For this purpose we specifically look at the probabilities that correspond to instances in which clients are waiting. Thus for clients $i = 1, \dots, n-s$:

$$\boldsymbol{p}_{\text{wait}}^{(s+i)}(t) := \Big(\boldsymbol{p}_{s+i}^{(s+i)}(t), \boldsymbol{p}_{s+i-1}^{(s+i)}(t), \dots, \boldsymbol{p}_{s+1}^{(s+i)}(t)\Big), \tag{3.16}$$

these probabilities adhere to the recursion earlier, and naturally one can define a start vector on arrival by $\boldsymbol{\alpha}_i^{\text{wait}} = \boldsymbol{p}_{\text{wait}}^{(s+i)}(0)$. Furthermore, the transitions between these probabilities over time are described by $\boldsymbol{S}_i^{\text{wait}}$ as defined in Eq. (3.14):

$$F_{W_{s+i}}(t) = 1 - \boldsymbol{p}_{\text{wait}}^{(s+i)}(t)\mathbf{1} = 1 - \boldsymbol{\alpha}_i^{\text{wait}} \exp\left(\boldsymbol{S}_i^{\text{wait}}\, t\right)\mathbf{1},$$

where $\mathbf{1}$ is a column vector of appropriate size. Also, for phase-type distributions the moments can readily be obtained by using its representation, so that

$$\sum_{i=s+1}^{n} \mathbb{E}W_i = \sum_{i=1}^{n-s} -\boldsymbol{\alpha}_i^{\text{wait}}(\boldsymbol{S}_i^{\text{wait}})^{-1}\mathbf{1}. \tag{3.17}$$

For idle and overtime, define $F_{M_{s+i}}(t)$ as the cumulative distribution function of the makespan of finishing the first $s+i$ clients t time units after t_{s+i} ($i = 1, \dots, n-s$), this is given by

$$F_{M_{s+i}}(t) = p_0^{(s+i)}(t) = 1 - \boldsymbol{p}^{(s+i)}(t)\,\mathbf{1} = 1 - \boldsymbol{\alpha}_{s+i}\exp\left(\boldsymbol{S}_{s+i}\, t\right)\mathbf{1},$$

so that $\mathbb{E}M_{s+i} = -\boldsymbol{\alpha}_{s+i}\boldsymbol{S}_{s+i}^{-1}\mathbf{1}$. In particular, the makespan corresponding to having all n clients served demarcates the session, and so the sum of all idle times can be described as the total time available in the system minus time spent in service. Similarly, overtime is incurred for all servers if a client remains in service after the targeted session-end time, T. Thus metrics for the servers' idle times, respectively overtimes, are given by

$$\mathbb{E}I^{(s)} = s\left(\mathbb{E}M_n + t_n\right) - \sum_{i=1}^{n} \mathbb{E}B_i; \tag{3.18}$$

$$\mathbb{E}O^{(s)} = s\int_0^\infty \max\{t + t_n - \mathrm{T}, 0\}\,\mathrm{d}F_{M_n}(t). \tag{3.19}$$

The targeted session-end time in the multi-server case is set to the sum of mean service times divided by the number of servers, as such it becomes equivalent to the single-server case. Note that the performance measures are superscripted, as to indicate that in a multi-server, setting idle time and overtime are incurred by all servers until the last server finishes the last client.

3.4.3 Convexity

For the single-server appointment scheduling problem in continuous time, strong stochastic convexity arguments can be invoked to prove that the waiting times are convex in the inter-arrivals. This follows from the fact that the Lindley recursion of Eq. (3.2) consists of convex operators, see Theorem (2.15) of Shanthikumar and Yao (1991). Other performance metrics can be expressed as convex functions of waiting times to establish convexity of the objective function (3.1), see for example Wang (1993) and Kuiper et al. (2023) for a proof that does not rely on stochastic convexity arguments.

For the multi-server queue, recursive systems exist which keep track of the workload per server, e.g. Kiefer and Wolfowitz (1955). Unfortunately, not all operators in this system are convex. As a consequence strong stochastic convexity arguments can not be applied to show convexity for general service-time distributions. Indeed, Harel (1990) found a counterexample which shows in stationarity that expected waiting times as a function of the inter-arrival time in the D/G/s queue are not convex, using a *bimodal* service time distribution: with $^2/_3$ probability the service-time equals 5 and with $^1/_3$ it equals 11 time units.

For phase-type distributions it is widely known that they can be used to approximate any non-negative distribution arbitrarily closely. The counterexample of Harel (1990) can easily be replicated by using a combination of two Erlang distributions with appropriate means, and many phases. Hence waiting times are not convex in inter-arrival times for the subclass D/PH/s either.

However, our phase-type distributions are unimodal, and we have strong reason to believe that our solutions are global optima as various starting points led to the same solutions. Finally, as considered in Section 3.6, we show that the optimization problem considered in the heavy-traffic regime, to which in essence many of the problems converge, is convex.

For the discrete analogue of the single-server appointment scheduling problem Zacharias and Yunes (2020) establish multi-modularity to guarantee that a global minimum is found. Our methodology can be used to show that multimodularity does not extend to the multi-server setting. For this purpose consider a schedule of equal slots, the first s clients arrive at the first slot starting at time zero and all subsequent clients arrive according to the equidistant schedule $x_{s+i} = {}^{\mathbb{E}B}\!/_{s}$. Then, choosing $n = 10$ and considering phase-type distributed service times with $\textsc{scv} = 1.5$ and number of servers $s = 2, 3, 4$, or $\textsc{scv} = 0.75$ and number of servers $s = 3, 4$ we find that neither the expected idle times of Eq. (3.18) nor the ones corrected for *early leave* (see Eq. (3.22) in Section 3.5.2) adhere to the first property as stated in Lemma 1 of Zacharias and Yunes (2020). Indeed, upon inspection, the proofs of multimodularity of the single-server performance metrics rely on keeping track of the workload per slot by means of the Lindley recursion, but this principle fails, as the variables for clients' waiting times and servers' workload do not coincide in a multi-server setting (Daley 1998).

3.5 Multi-Server Appointment Scheduling in Transient Settings

In this section, we employ our methodology to compute appointment schedules for the multi-server setting and contrast that with implementing a complementary set-up of optimized single-server appointment schedules. In addition, the impact of service-time variability, no-shows and some relevant modifications to the objective functions, such as early leave of servers, are studied.

In our analyses we examine the optimal solutions found when appointments are scheduled for multiple servers. For our computations we relied on a standard machine and our programs are written in `MATLAB`. For minimization `MATLAB`'s built-in routine `fmincon` is employed. We use the metrics as defined in Section 3.4.2 and set the cost of waiting time to one, $c_W = 1$, so that the cost ratio of idle to waiting time simplifies to c_I. Hence, our minimization becomes

$$\min_{(x_{s+1},\dots,x_n)} c_I\,\mathbb{E}I^{(s)} + \sum_{i=s+1}^{n} \mathbb{E}W_i + c_O\,\mathbb{E}O^{(s)}, \tag{3.20}$$

so that the non-trivial arrival epochs follow from $t_{s+i} = \sum_{j=1}^{i} x_{s+j}$.

The explicit expressions for the metrics given in Section 3.4.2 facilitate computation. Note that if $c_O = 0$, the objective function reduces to a closed-form expression. Research by Klassen and Yoogalingam (2014) suggests that including overtime into the objective function has roughly the same impact as increasing the weight put on idle time. As a consequence most of our experiments concentrate on the case of idle and waiting time only, although in Section 3.5.2 we specifically study the inclusion of overtime. We verified the optimizations by choosing as different starting points vectors consisting of only zeros, ones, average service rates (${}^{1}\!/_{s}$), and the heavy-traffic solution as obtained in Section 3.6.2.

3.5.1 Structure of the Optimal Solution

Studying the patterns of the optimal inter-arrival times of multi-server appointment schedules for various numbers of clients, the characteristic dome-shape plateau to which solutions converge can be identified. For example, in Figure 3.1, for $s = 1$ clear dome shapes appear. For the pooled schedules, so where $s = 2$, 3 or 4, the middle solutions converge to steady-state values, encompassing the long-term balance between idling and waiting. Note that because the number of clients to be scheduled is fixed, there is one inter-arrival fewer to be determined if the number of servers s increases.

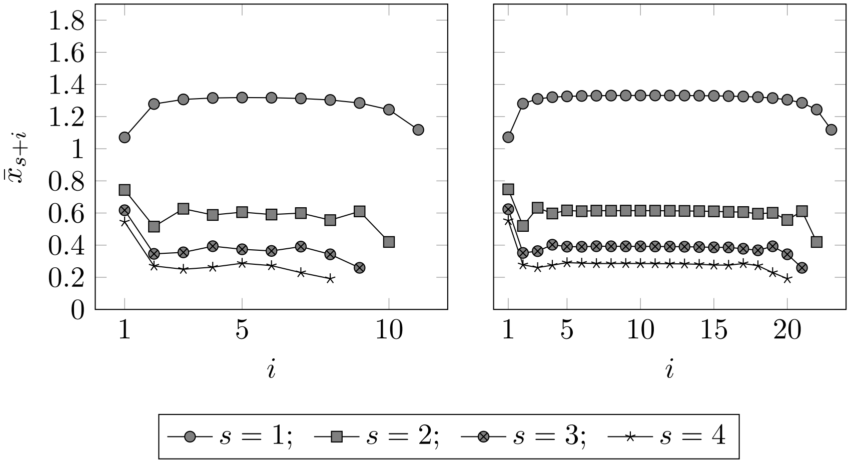


Figure 3.1: In both panels SCV = 0.25 (approximated by *mixture of Erlang* service times) and $c_I = 1$, left there are 12 clients to be scheduled whereas in the right panel 24. The graphs in each panel show the pattern of inter-arrival times, wherein from top to bottom the number of servers is increased from one to four.

At the start of a session, as seen in Figure 3.1 and also in Figure 3.2, we observe a *reversed bullwhip* in the inter-arrival times; a steep decline in the inter-arrival times is followed by a damping pattern of iteratively increasing and decreasing inter-arrival times. The reason for this pattern is that the synchronized start of service is completely absorbed by the randomness in the system if there are sufficient clients to be scheduled. Comparing Figures 3.1 and 3.2, the extent of this effect is amplified for lower values of SCV. In the extreme case of no uncertainty (i.e., the D/D/s queue) the optimal schedule for s servers would be the arrival of a batch of s clients after each mean service time.

If SCV equals one these patterns disappear, see Figure 3.3, possibly due to *memorylessness* of the exponential service times that are used to model the service times: aside from the number of clients in the system at each arrival no additional information is revealed about how far along each service time is. In case of SCV greater than one, a mixture of exponential service times is used,

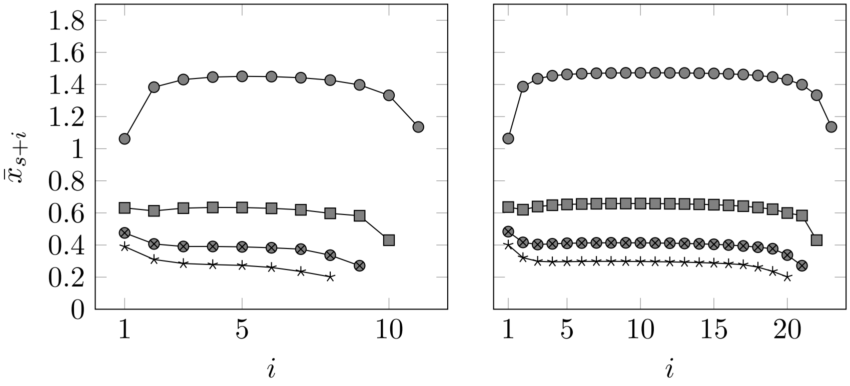


Figure 3.2: In both panels SCV = 0.5 (approximated by *mixture of Erlang* service times) and $c_I = 1$, left there are 12 clients to be scheduled whereas in the right panel 24. The graphs in each panel show the pattern of inter-arrival times, wherein from top to bottom the number of servers is increased from one to four, cf. legend Figure 3.1.

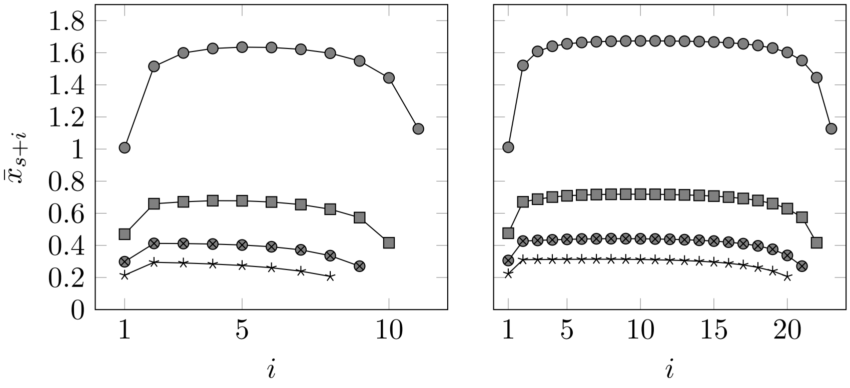


Figure 3.3: In both panels SCV = 1 (approximated by *exponential* service times) and $c_I = 1$, left there are 12 clients to be scheduled whereas in the right panel 24. The graphs in each panel show the pattern of inter-arrival times, wherein from top to bottom the number of servers is increased from one to four, cf. legend Figure 3.1.

which contains even greater variability than the exponential case, and so as in Figure 3.4 the dome-shape pattern stands out.

Focusing on the session end, fluctuating inter-arrival times are apparent in a bullwhip pattern for low SCV values. In particular, analogously to the start-up, these patterns are stronger for lower SCV cases, and evidently the effect disappears for SCV ≥ 1. The explanation for this behavior is that unused capacity on other servers is penalized as idle time, so the optimization tries to synchronize the servers' end times at the expense of the desired reduction in waiting times. Waiting time becomes less important towards the end of the session as there will be fewer clients who would be affected by a tight schedule.

3.5.2 Session-End Revisited

One can opt to include session overtime in the framework, which is computed by Eq. (3.19). In Figure 3.5 we report the optimized inter-arrival times that result from an objective function that is composed of only waiting time and overtime. Here we choose $c_O = 1.5$, 1.5 times the value chosen for c_I in Figure 3.2, which is typical as argued in Çayırlı et al. (2012). In Figure 3.5 we added to the optimal solutions the solutions from Figure 3.2, which incorporated idle time instead of overtime. Remarkably, for any number of servers a comparison of the shape of the curves reveals that including overtime has a similar impact as idle time, which echoes the conclusions of Klassen and Yoogalingam (2014) for the single-server case.

Another salient feature of a multi-server appointment schedule is that servers can finish earlier when there is insufficient work left, that is there are fewer clients in the process and in the appointment schedule than the number of servers. For example, in Figure 3.6(b) servers two and three can finish earlier. By allowing this early leave, the idle time of servers waiting until the last client has finished service can be reduced, this time is indicated by the diagonal lines. So far this feature is not incorporated in the objective function as it would cause unsynchronized endings to not be penalized, and thus lead to unnecessary underutilization of a session. Naturally, in a system of single-server appointment schedules a server leaves when there is no client left, as seen in the system in Figure 3.6(a). So in order to have a balanced comparison, also in terms of idle times, between a system of single-server systems and a multi-server setting a correction for early leave of servers is necessary.

After optimizing over the objective function in display (3.20), the additional gain of allowing early leave of servers can be computed. Obviously a server can only finish if it is certain that the server will not be required in the future. So if there are ξ servers busy ($\xi \in \{1, 2, \ldots, s\}$) and one finishes, it can be released if and only if there are exactly $\xi-1$ clients remaining to be served, that is those currently in service plus the ones still scheduled. To highlight the subtlety, note that the third server in Figure 3.6(b) can only leave after the 10-th client has left and no sooner, which is after the *cross with dots*. At that time there is one client yet to *start* and one still *in* service.

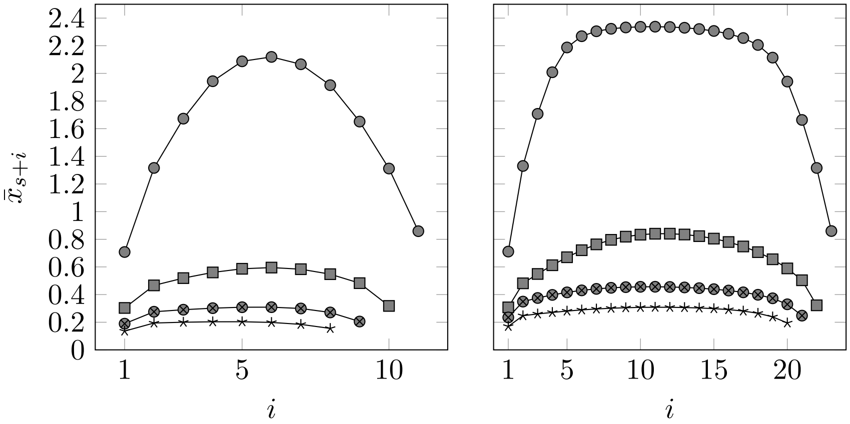


Figure 3.4: In both panels SCV = 4 (approximated by *hyperexponential* service times) and $c_I = 1$, left there are 12 clients to be scheduled whereas in the right panel 24. The graphs in each panel show the pattern of inter-arrival times, wherein from top to bottom the number of servers is increased from one to four, cf. legend Figure 3.1.

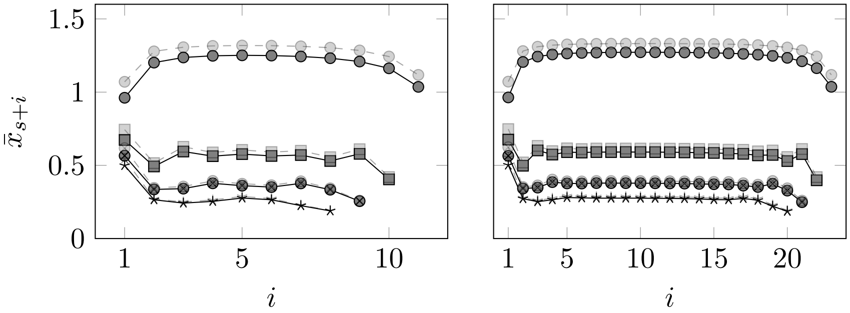


Figure 3.5: In both panels the fully opaque marks represent SCV = 0.25 and $c_O = 1.5$ ($c_I = 0$), which contrast to the semi-transparent marks originally from Figure 3.1 in which $c_I = 1$ ($c_O = 0$). Left there are 12 clients to be scheduled whereas in the right panel 24. The graphs show the pattern of inter-arrival times, wherein from top to bottom the number of servers is increased from one to four, cf. legend Figure 3.1.

The expected finish time of a server in a pooled system (numbered in reversed order of leaving) can be computed by constructing elements of a cumulative distribution function. The ℓ-th server ($\ell \in \{1, \ldots, \xi\}$) can finish in a time $t \in [0, x_{n-\ell+j+1})$ after the $(n{-}\ell{+}j)$-th arrival for $j \in \{1, \ldots, \ell\}$ with $x_{n+1} := \infty$, so there are $\ell - j$ clients yet to arrive, if there are fewer than j clients in service. This leads to

$$F_{E_{\ell,j}}(t) = 1 - \sum_{i=j}^{n-\ell+j} \boldsymbol{p}_i^{(n-\ell+j)}(t)\,\mathbf{1},$$

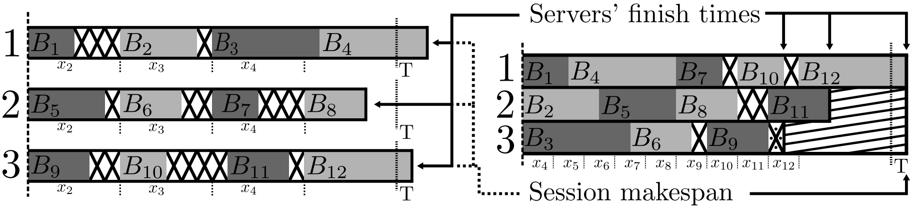
recall that $\boldsymbol{p}_i^{(n-\ell+j)}$ describes the phases after the arrival of the $(n{-}\ell{+}j)$-th client for which i clients remain in the system, see Eq. (3.7). The expected finish time of server ℓ and server specific overtime can be computed via the numerical integration of:

$$\begin{aligned} \mathbb{E}E_\ell &= \sum_{j=1}^{\ell} \int_0^{x_{n-\ell+j+1}} (t + t_{n-\ell+j})\,\mathrm{d}F_{E_{\ell,j}}(t); \\ \mathbb{E}O_\ell &= \sum_{j=1}^{\ell} \int_0^{x_{n-\ell+j+1}} \max\{t + t_{n-\ell+j} - \mathrm{T}, 0\}\,\mathrm{d}F_{E_{\ell,j}}(t). \end{aligned} \tag{3.21}$$

The session-end metrics defined in Eqs. (3.18) and (3.19) relate accordingly: $\mathbb{E}I^{(s)} \equiv s\,\mathbb{E}E_1 - \sum_{i=1}^{n} \mathbb{E}B_i$ and $\mathbb{E}O^{(s)} \equiv s\,\mathbb{E}O_1$. Since servers leave when no longer needed, the expected idle times throughout the schedule are computed by

$$\sum_{i=s+1}^{n} \mathbb{E}I_i = \sum_{\ell=1}^{s} \mathbb{E}E_\ell - \sum_{i=1}^{n} \mathbb{E}B_i. \tag{3.22}$$

This re-visitation of the session end extends Eq. (3.3) to the multi-server setting, allowing a comparison to equivalent systems of single servers, i.e., to study the impact of pooling.



(a) Single servers

(b) Pooled servers

Figure 3.6: A visualization of two appointment systems which serve 12 clients on three servers; on the left singly operating servers (each with the same schedule) and on the right in parallel. The *crosses* indicate idle time; the *diagonal lines* the gain won by allowing early leave; B_i are the service times and T the session-end time.

3.5.3 Benefits of Pooling

Besides studying the structure of the optimal solutions we are also interested in a comparison of performance between having server-dedicated appointment schedules versus pooled, multi-server appointment schedule. As reported for call centers (e.g., van Dijk and van der Sluis 2008), we anticipate that the pooling of resources will be highly beneficial.

	$n = 12$			$n = 24$			$n = 48$		
Expected costs in an optimized system of s single servers *(a)*									
s	$\sum \mathbb{E}I_i$	$\sum \mathbb{E}W_i$	$\sum \mathbb{E}O_\ell$	$\sum \mathbb{E}I_i$	$\sum \mathbb{E}W_i$	$\sum \mathbb{E}O_\ell$	$\sum \mathbb{E}I_i$	$\sum \mathbb{E}W_i$	$\sum \mathbb{E}O_\ell$
2	3.4118	3.4602	3.4118	8.9153	7.2742	8.9153	20.2858	14.6852	20.2858
3	2.6000	3.1740	2.7369	7.8055	7.1316	7.8055	19.0337	14.6337	19.0337
4	1.9706	2.8264	2.5451	6.8236	6.9204	6.8236	17.8307	14.5483	17.8307
Expected costs in an optimized system of s pooled servers *(b)*									
2	2.4315	2.3272	2.4415	6.2922	4.8752	6.2922	14.2322	9.8381	14.2322
3	1.5421	1.6861	1.8252	4.5390	3.7538	4.5393	10.9165	7.6884	10.9165
4	1.0314	1.2768	1.7097	3.4807	3.0745	3.5521	8.8986	6.4358	8.8986
Performance gains ($^{a-b}/_a$)									
2	28.73%	32.74%	28.44%	29.42%	32.98%	29.42%	29.84%	33.01%	29.84%
3	40.69%	46.88%	33.31%	41.85%	47.36%	41.84%	42.65%	47.46%	42.65%
4	47.66%	54.83%	32.82%	48.99%	55.57%	47.94%	50.09%	55.76%	50.09%

Table 3.2: In these experiments SCV is set to 0.5 and $c_I = 1$, i.e., idle and waiting time are valued equally importantly. Overtimes are computed after optimization with the aforementioned settings using $\mathrm{T} = {}^n/_s$.

To understand the merits of pooling, we compare the performance of our multi-server appointment schedules to those in which an equivalent system of single-server schedules are employed. In order to have a balanced comparison we compute the expected overtimes per server and idle times throughout the session by using Eqs. (3.21) and (3.22) after the optimization, as to examine how an individual server benefits from being in a pooled system. Varying the number of clients in multiples of 12, divisible by two, three or four (servers), we report the expected performance in Table 3.2.

The performance improvement is striking, and significant reductions in each performance dimension appear; up to around 55% when four servers are pooled. Note that in some cases the sum of overtimes and idle times are the same, which naturally occurs when the last client is scheduled after the targeted session-end time: $\sum_{i=1}^{n-s} \bar{x}_{s+i} = t_n > \mathrm{T}$.

In appointment scheduling, the servers' time is often considered to be more valuable than that of clients. In healthcare, for example, waiting time is valued considerably less than idle time (Robinson and Chen 2011). Therefore we experiment here with settings in which the cost parameter c_I takes a high value in the objective (3.20). Besides depicting the baseline schedule of Figure 3.1, wherein $c_I = 1$, the optimal schedules are shown in Figure 3.7 for $c_I = 5$, in

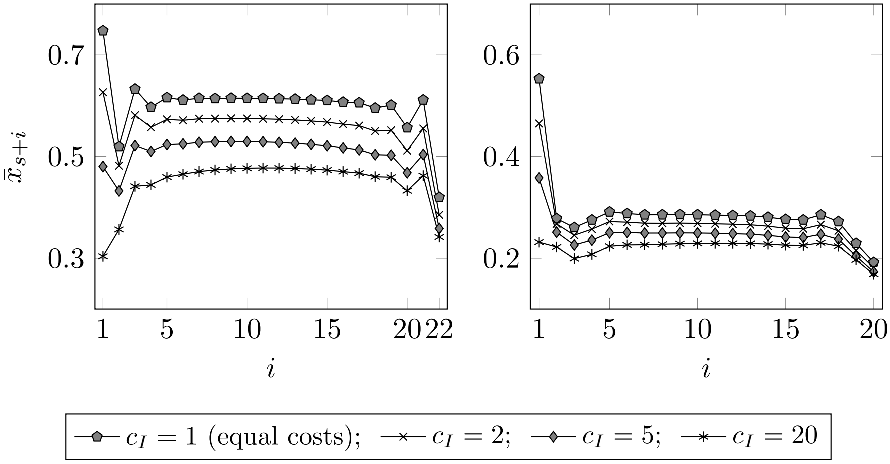


Figure 3.7: Extending on the setting of Figure 3.1, $n = 24$ and SCV $= 0.25$, the optimal appointment schedules when pooling two (left) or four (right) servers whilst varying c_I in the objective function of display (3.20).

accordance with the middle setting of Çayırlı et al. (2012), and $c_I = 20$ as an extreme case, albeit in line with the observations of Robinson and Chen (2011). Besides moving the dome-shape pattern down, i.e., tightening the schedule, placing a lower value on waiting time damps the distinctive start and end patterns of a multi-server schedule.

Considering the gains achieved by pooling in Table 3.3 in the case of $c_I = 5$, we observe that the improvements on the servers' account lag behind in the smaller instances. This effect is due to the imbalanced effort that is now put on reducing idling, and consequently overtime, resulting in tight schedules in the corresponding system of s single servers. Contrariwise the expected waiting times decrease greatly, see Table 3.2. For $n = 48$, moving away from dominating transient effects, we conclude that the performance improvement for idle and overtime mimics those reported earlier. This is backed by our analysis in Section 3.6 wherein the performance improvement in the long run turns out to be a factor of $\sqrt{s}$.

3.5.4 No-Shows

No-shows are recognized as an important environmental factor to be accounted for in appointment schedules (e.g., Ho and Lau 1992, Çayırlı and Veral 2003) and are also studied in detail for multi-server systems in Zacharias and Pinedo (2017). To accommodate for the impact of no-shows, which occur with probability q, in our framework, we only have to adapt the initial probability vectors. With probability $(1-q)$ we have a client whose service time is approximated by a phase-type distribution and with probability q a client who does not show-up at all.

	$n = 12$			$n = 24$			$n = 48$		
Expected costs in an optimized system of s single servers *(a)*									
s	$\sum \mathbb{E}I_i$	$\sum \mathbb{E}W_i$	$\sum \mathbb{E}O_\ell$	$\sum \mathbb{E}I_i$	$\sum \mathbb{E}W_i$	$\sum \mathbb{E}O_\ell$	$\sum \mathbb{E}I_i$	$\sum \mathbb{E}W_i$	$\sum \mathbb{E}O_\ell$
2	0.7593	9.1779	1.5273	2.6319	21.3441	2.8663	7.3178	44.2482	7.3178
3	0.4742	7.6170	1.7530	1.9720	19.9379	2.8714	6.1683	43.7249	6.2061
4	0.3085	6.2159	1.9732	1.5185	18.3558	3.0546	5.2639	42.6882	5.7325
Expected costs in an optimized system of s pooled servers *(b)*									
2	0.5876	6.3901	1.1895	1.9504	14.7088	2.1586	5.2899	30.3613	5.2899
3	0.3306	4.3396	1.2471	1.2623	11.1103	1.8984	3.7485	24.1044	3.8288
4	0.2042	3.0992	1.3759	0.8922	8.8466	1.8916	2.8589	20.2124	3.2533
Performance gains ($^{a-b}/_{a}$)									
2	22.62%	30.38%	22.12%	25.89%	31.09%	24.69%	27.71%	31.38%	27.71%
3	30.29%	43.03%	28.86%	35.99%	44.28%	33.89%	39.23%	44.87%	38.31%
4	33.81%	50.14%	30.27%	41.25%	51.80%	38.07%	45.69%	52.65%	43.25%

Table 3.3: In these experiments SCV is set to 0.5 and $c_I = 5$, i.e. idle time is valued as five times more important than waiting time. Overtimes are computed after optimization with the aforementioned settings using $\mathrm{T} = {}^{n}/_{s}$.

Define $\boldsymbol{\alpha}^q_{s,j} = (\boldsymbol{\alpha} \otimes \cdots \otimes \boldsymbol{\alpha})(1-q)^j q^{(s-j)} \binom{s}{j}$, where the Kronecker product is applied exactly j times. Now the start vector reads $\boldsymbol{\alpha}^q_s = (\boldsymbol{\alpha}^q_{s,s}, \boldsymbol{\alpha}^q_{s,s-1}, \ldots, \boldsymbol{\alpha}^q_{s,1})$, and when a subsequent client *should* arrive, the lines in Eq. (3.15) that define the recursion are replaced by:

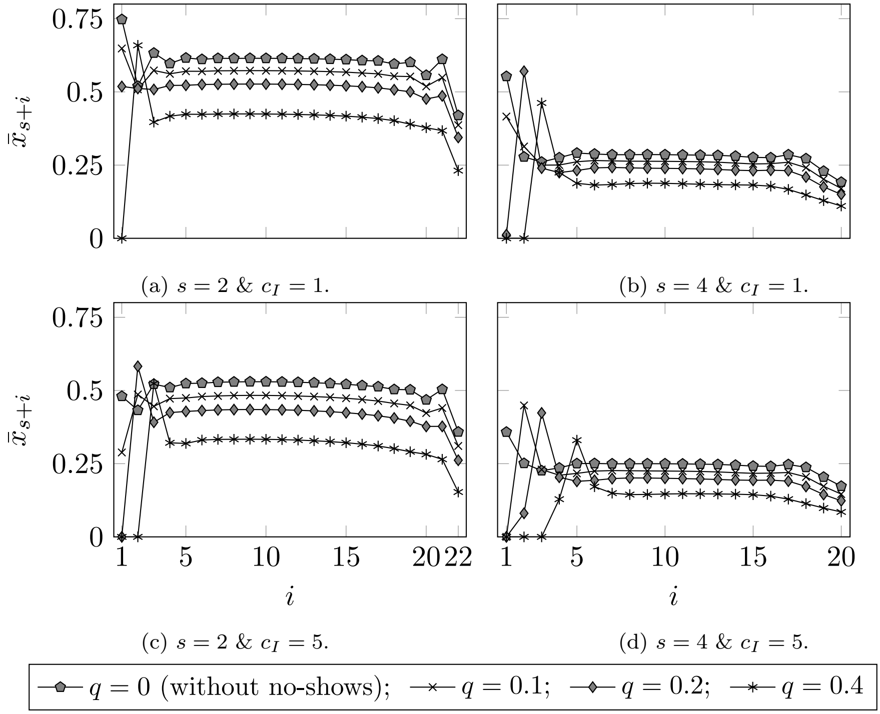
$$\boldsymbol{\alpha}^q_{s+i} = (1-q)\left(f\left(\boldsymbol{p}^{(s+i-1)}(x_{s+i}), \boldsymbol{\alpha}\right)\right) + q\left(\boldsymbol{0}_{k^s}, \boldsymbol{p}^{(s+i-1)}(x_{s+i})\right) \tag{3.23}$$

using the $f(\cdot, \boldsymbol{\alpha})$ function as also defined in Eq. (3.15). Analyzing Eq. (3.23) shows that with probability $(1-q)$ the $(s+i)$-th client is added to the system, either in service or in the queue, and with probability q the system remains unchanged as the client did not show up. Using the earlier transition matrices, the possible transitions remain unchanged, we conclude that $\mathrm{PH}(\boldsymbol{\alpha}^q_{s+i}, \boldsymbol{S}_{s+i})$ describes the system after the $(s+i)$-th client's arrival on which we apply the developed machinery; Eq. (3.22) should be adapted accordingly to account for no-shows by subtracting $(1-q)\sum_{i=1}^{n} \mathbb{E}B_i$ instead.

On top of the baseline setting of Figure 3.1, we implemented no-shows to occur with probabilities 10%, 20%, and 40% in Figure 3.8 for different numbers of servers and cost parameters. The no-show levels chosen cover the range reported in Çayırlı et al. (2012). Due to the occurrence of no-shows, a scheduled client effectively brings in less work, so we observe that the dome is pushed downwards by approximately the fraction with which no-shows occur. This happens irrespectively of the weight chosen for idle time in the objective function, c_I.

More interestingly, with no-shows we see that at the beginning the optimization counter-acts the possibility of server idling; the first inter-arrival(s) decrease and even become zero, which means starting a session with more clients scheduled than the number of servers s. This overbooking will typically be followed by a relatively high inter-arrival time, as several panels in Figure 3.8 clearly show. In single-server equivalents an overbooking to the first slot only occurs at a no-show probability of 40%, after which the dome-shape pattern commences. Overbooking is more persistent when more servers are pooled as for multiple servers it hedges against the possibility of idling without costing more in terms of waiting time, since the queue is serviced by more than one server.

At the end of the schedules in Figure 3.8, we see that no-shows dampen the idiosyncratic fluctuations. Comparing the top to the bottom panels, we observe that the schedule has tightened, and overbooking has occurred at more occasions and with greater severity. This is the result of valuing idle time more importantly in the objective function. Still, we see that also in these cases a relatively long inter-arrival time follows, demonstrating that the effect of no-shows cannot be dismissed.



(a) $s = 2$ & $c_I = 1$.

(b) $s = 4$ & $c_I = 1$.

(c) $s = 2$ & $c_I = 5$.

(d) $s = 4$ & $c_I = 5$.

Figure 3.8: Extending on the setting of Figure 3.1, $n = 24$, SCV $= 0.25$, with varying no-show probability q for different numbers of pooled servers s and cost parameter c_I.

3.6 Multi-Server Appointment Scheduling in Steady-State Settings

As noted in Çayırlı and Veral (2003) steady-state is never reached in a real clinical session with a small number of clients. However, as seen in the various figures that depict the optimal transient schedule the middle-most values converge quickly to a stationary plateau. Therefore, the steady-state counterparts of these systems provide relevant insight for the transient setting and are appropriate for the majority of clients. Indeed, studying the clinic in steady state has lead to a fruitful stream of research (Lindley 1952, Jansson 1966, Swisher et al. 2001, Kuiper et al. 2017).

The reason why the plateau of each dome converges to the corresponding optimal steady-state inter-arrival time is intuitive, since transient effects found at the start and end of a session favour service providers. Therefore waiting time is valued less in a transient setting. In steady state these transient effects are neglected, as the session has run forever, so that the extent of waiting time propagates to possibly all clients, which results in the optimal stationary inter-arrival time bounding the plateau of the dome from above.

In a stationary analysis the transient effects at the start and toward the end of the session are neglected. As a consequence, the objective function reduces to an elegant combination of idle time, which can be expressed as excessive capacity $sx - \mathbb{E}B$, versus clients' waiting times $\mathbb{E}W$. Let x be the steady-state inter-arrival time then, given without loss of generality $c_W \equiv 1$, we find:

$$\arg\min_x c_I \, \mathbb{E}I(x) + c_W \, \mathbb{E}W(x) = \arg\min_x c_I \, sx + \mathbb{E}S(x), \tag{3.24}$$

with $S(x)$ the sojourn-time distribution that obviously depends on the stationary inter-arrival time x. Given the fact that we minimize an objective function with $c_I < \infty$ the utilization will never reach a fully loaded system so that the existence of a steady-state solution is guaranteed (Kiefer and Wolfowitz 1955, Whitt 1982). In the next sections we propose two methods that provide solutions in this limiting regime.

3.6.1 Phase-Type Approach

The transition matrix that is obtained in Section 3.5 can be used to compute the equilibrium distribution, $\boldsymbol{\pi}$, by considering the system's embedded Markov chain. Consider the system slightly before a new client's arrival, then the system jumps to a state with an additional client, then after x amount of time the system should have returned to its equilibrium distribution. The full system contains an infinite number of states, but since the probability of having n clients in the system goes rapidly to zero as n grows large, we cut-off the number of clients to be allowed in the system at n; in our experiments $n = 40$ has worked well. This allows the computation of the steady-state distribution for a given x by solving:

$$\boldsymbol{\pi}_0^{(n)} = f\left(\boldsymbol{\pi}^{(n)}, \boldsymbol{\alpha}\right) \boldsymbol{P}_n, \quad \text{where } \boldsymbol{P}_n = \left[\exp(\boldsymbol{S}_{n+1}x), 1 - \exp(\boldsymbol{S}_{n+1}x)\mathbf{1}\right],$$

with $\boldsymbol{\pi}_0^{(n)} := \left(\boldsymbol{\pi}^{(n)}, \pi_0^{(n)}\right)$ and thus $\boldsymbol{\pi}^{(n)}$ having similar states as $\boldsymbol{p}^{(n)}(x)$ of Eq. (3.12), so that the function $f(\cdot, \boldsymbol{\alpha})$ as defined in Eq. (3.15) can be applied. Subsequently the transition matrix extended with the transitions to the empty state can be used to obtain the steady-state probabilities for each of the states in the vector $\boldsymbol{\pi}_n$ by solving the above system; cutting off the transitions to states with $n+1$ clients and imposing the normalization equation: $\boldsymbol{\pi}_0^{(n)}\mathbf{1} = 1$.

This non-singular system for the steady state can be solved by exploiting, e.g., `MATLAB`'s built-in routines to compute the matrix exponential and solve the system of linear equations. Then $\boldsymbol{\pi}_n$ can be used as the initial probability vector $\boldsymbol{\alpha}_n$ and the transition matrix is just $\boldsymbol{S}_n$, so that the performance metrics of Section 3.4.2 can be computed and filled into the objective function of (3.24). Optimization over x provides us the optimal stationary inter-arrival time $\bar{x}_{\text{pt}}$.

In Figure 3.9, where an equal cost ratio is chosen ($c_I = 1$), we observe a decreasing pattern in the expected idle and waiting times when the number of servers increases. As seen, the marginal decreases in idle and waiting times are less for higher values of s, which is in line with the theoretical result for waiting times in G/G/s queues derived in Weber (1980). Furthermore, we see that the performance gain achieved by pooling is greater when the SCV is larger; SCV varies from 0.5 (reflecting healthcare environments) up to 1 (exponential service times).

In the next section we make an interesting connection with appointment scheduling studied in a heavy-traffic regime and compare the stationary solutions obtained by the phase-type approach with those obtained under heavy traffic for which elegant expressions are derived.

3.6.2 Robust Schedules

The goal in many appointment scheduling problems is to reduce waiting time while keeping utilization at a high level, so that no capacity is wasted. Consequently, the idle time should be low so that the load of the system is close to 1. This observation warrants consideration of the problem in a heavy-traffic regime. In our cases, this entails a *steady-state* inter-arrival time x only slightly larger than the mean service time divided by the number of servers, i.e., $^{\mathbb{E}B}/_{s}$.

In fact, when s increases, our steady-state results using the method outlined above converge to the results obtained in the heavy-traffic regime. Because the variability accrued is spread over multiple servers, expected waiting and idle times will be lower than in an equivalent system of single servers. Consequently, higher utilizations are achieved; the inter-arrival times are much closer to the service rate. Moreover, as for the single-server case, as idle time is evaluated more importantly, inter-arrival times tend to the service rate, see also

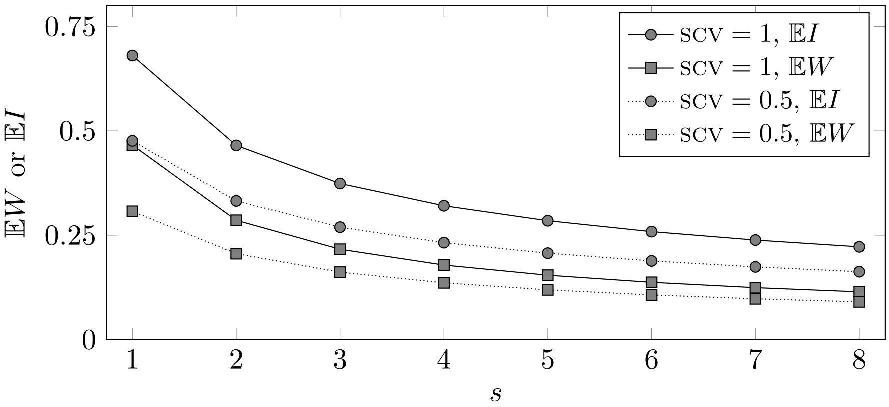


Figure 3.9: The expected waiting and idle times corresponding to the optimal stationary solutions for $s \in \{1, \dots, 8\}$ using the embedded Markov chain from the phase-type approach; $c_I = 1$, while SCV is set to 0.5 and 1.

our numerical results in the previous section. So it should generally hold that in these two scenarios a heavy-traffic approximation will provide an accurate approximation.

Using the steady-state result for the G/G/s under a heavy-traffic regime, see for example *Theorem 2* in Köllerström (1974) or Section 5 in Whitt (1983), we have

$$\frac{2s(sx - W)}{\text{SCV}} \sim \text{Exp}(1), \quad \text{when} \quad x \downarrow {}^{\mathbb{E}B}/_{s}.$$

Specifically, since the means are normalized, we can rewrite the optimization problem of (3.24) for given c_I, which is easily solved using straightforward calculus:

$$\bar{x}_{\text{ht}} = \frac{1}{s}\left(1 + \sqrt{\frac{\text{SCV}}{2c_I s}}\right) = \underset{x \in ({}^{\mathbb{E}B}/_{s}, \infty)}{\arg\min}\; c_I(sx - 1) + \frac{\text{SCV}}{2s(sx-1)}.$$

Furthermore, since the second derivative in x is positive, a global optimum is guaranteed. We observe that the margin to account for randomness on top of the average service rate is a multiplication of ${}^{1}/_{\sqrt{s}}$, which tends to zero as s increases.

Studying the optimal solution as a function of the inter-arrival times, we observe in Figure 3.10 that for c_I values greater than 1, given that SCV is not too large, the heavy-traffic and the solutions obtained by the phase-type approximation are nearly the same. In fact, the graphs depicted in Figure 3.11 show that for low SCV values, say SCV < 1, heavy-traffic solutions provide accurate approximations even when c_I equals 1.

In addition, the corresponding expected idle and waiting times provide insight into the patterns observed in Figure 3.9, and are explicitly given by:

$$\mathbb{E}I_{\text{ht}} = \sqrt{\frac{\text{SCV}}{2s\, c_I}} \quad \text{and} \quad \mathbb{E}W_{\text{ht}} = \sqrt{\frac{\text{SCV}\, c_I}{2s}}. \tag{3.25}$$

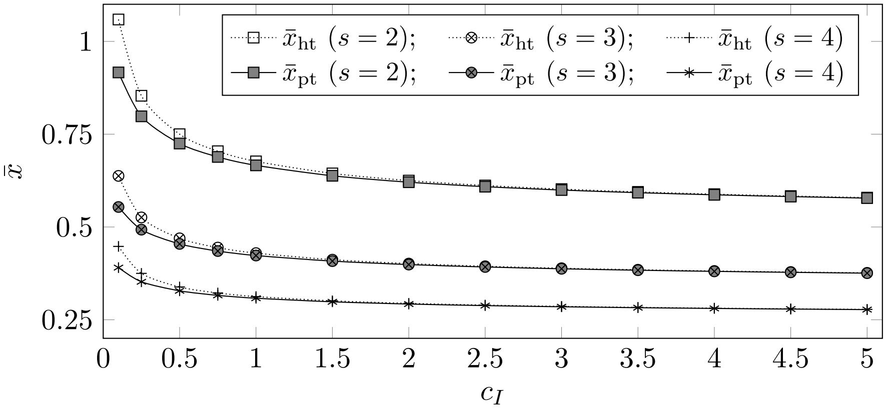


Figure 3.10: For SCV = 0.5 the optimal stationary solutions for a wide range of c_I values, cost ratios vary from 5:1 to 1:10, using the phase-type approach $\bar{x}_{\text{pt}}$ or the heavy-traffic approximation $\bar{x}_{\text{ht}}$.

For a decision-maker these expressions reveal how the operational benefits of pooling can be concentrated on either one of the performance dimensions. If utilization is the primary concern, idle times can even be reduced by a factor of s, keeping waiting times the same, by multiplying c_I with s when servers are pooled. Conversely, when waiting times are of paramount importance c_I should be divided by s.

Note that the set-up discussed here is robust against misspecification of the distribution and only depends on the first two moments. Moreover the waiting times in heavy-traffic coincide with the conjectured upper bound on the waiting times for multi-server queues (Daley 1998), equating the heavy-traffic results to that obtained whilst minimizing the objective function under the worst-case distribution, cf. Mak et al. (2015).

3.7 Conclusion and Discussion

In this work the intensively studied single-server appointment schedule problem is extended to a multi-server setting. The multi-server setting introduces many obstacles to tracking and convexity that are not present when only one server is considered. Due to this fact, this case is not studied in an analytical manner in the literature, although it features in many service systems. We offer a computational approach to the multi-server setting relying on the tractability of phase-type distributions, which is employed to gain insight in the optimal solution in multi-server appointment schedules.

The impact of deviating from the class of phase-type distributions in the transient setting would be a logical avenue for further research. Furthermore it remains open whether in each instance a global optimum is found. By the

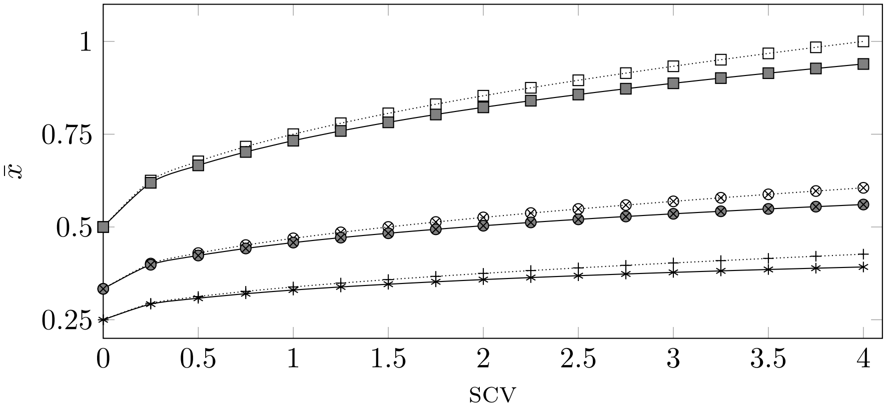


Figure 3.11: The optimal stationary solutions when $c_I = 1$ for a range of SCV values, using the phase-type approach or the heavy-traffic approximation; the solutions are differentiated according to the same legend as in Figure 3.10.

fact that our optimizations converged to the same solution when varying start vectors and that in corresponding heavy-traffic regimes convexity can be shown, we have strong reason to believe that the problems considered are convex. Of course this remains a challenging line of research for queueing theorists. Lastly, confining the solutions over discrete points in time might be an extension that might be of particular value to practice; such a model is studied for a single server in Zacharias and Yunes (2020).

A well-known result of the single-server appointment scheduling is the dome-shape pattern. Contrasting with the multi-server setting there are some discrepancies that arise at the start and the end of a session for cases in which SCV is below one; typical in many service systems such as in healthcare (Çayırlı and Veral 2003). These patterns arise due to multiple servers starting synchronously, each serving one client. The apparent pattern is characterized by a damped bullwhip, which converges in the middle to a steady-state plateau as the randomness in the service times gradually suppresses these effects. At the end of the session, the optimization tries to achieve a synchronous ending of the servers, which culminates in a similar but reversed pattern in the optimal inter-arrival times.

These patterns gradually decrease when SCV tends to one; if SCV equals one (or is higher), the exponential distribution (or a mixture) is used for which a dome-shape pattern appears for multiple servers as well. The inclusion of overtime has the same impact as incorporating idle time in the optimization. If idle time is valued more importantly, pooled appointment schedules are tightened and the atypical start and end of session patterns are damped. Experiments further reveal that including no-shows lowers the plateau by the fraction in which they occur and the striking multi-server patterns at the end are damped. At the start of the session, no-shows result in overbooking, which in the case of low SCV values is followed by an extremely large inter-arrival time. This idiosyncratic initialization behavior persists even for higher cost parameters.

In addition, a relevant selection of cases provides practitioners insight into the decisions and trade-offs that arise. In detail, the performance gains of pooled appointment systems versus equivalent systems of singly operating servers are analyzed in a framework that incorporates service-time variation. Focusing on the waiting times, the analysis shows that the expected waiting times reduce by about 31% when two servers are pooled, and for four servers by an astounding 53%. Similar double-digit reductions are reported for expected idle time and overtime. In healthcare, for example, the comparison unravels the implicit cost of continuity of care.

The optimal stationary schedule is also studied, which approximates the dome-shape plateau arising in the transient setting. Notice that with more servers the variation in the system is reduced and thus a heavy-traffic regime becomes an appropriate modeling framework. The optimal solution in this regime has an algebraic expression. Moreover, this regime elucidates that the expected idle or waiting times decrease by a factor of $\sqrt{s}$ when s servers are pooled. Finally, it is likely that the heavy-traffic solution coincides with the conjectured upper bound (Daley 1998) on the expected waiting time in a multi-server setting and is thus robust. As such, this study provides a comprehensive account of the multi-server appointment scheduling problem, which was as yet unaccounted for in the field.

Chapter 4

On Scheduling Multiple Patients

4.1 Introduction

There is a great variety of designs in manufacturing, communication, and service systems. We will distinguish in particular two queueing structures: a system operated by a single server, and one operated by multiple servers in parallel, in which case it is unclear a-priori by which server a customer will be served.

Combining multiple servers in parallel can be termed *pooling.* This pooling carries with it significant efficiency gains. See van Dijk and van der Sluis (2008) for an example in a call center context and Benjaafar (1995) for an overall performance analysis. However, pooling and its merits have received little attention in the sub-domain of appointment scheduling. Appointment scheduling is the specific domain in which customers arrive according to an appointment book. This domain has traditionally centered on the single-server setting, which is primarily motivated in the healthcare setting by a desire for continuity of care, where one patient will always see the same physician, but also by the fact that multi-server settings are intrinsically more complex to analyze.

Relaxing this assumption is in many scheduling situations not unreasonable, as described in the recent work by Soltani et al. (2019). Cases where this may occur are performing blood tests, inoculations, or when a customer needs only a single appointment, e.g., a COVID-test, see for example Hanly et al. (2021) where they analyse a queueing network for mass vaccination hubs, supposing that patients arrive singly at 10 minute intervals, or in batches of 120 within a 60-minute interval.

Even in situations where continuity of care does play a role, there are several studies pointing out that in practice this single-server paradigm is violated, such as Liu and Liu (1998) and Balasubramanian et al. (2010), who report that patients prefer seeing a different physician if this reduces their waiting time. Moreover, the recent COVID pandemic transformed many situations from ones were customers were free to choose their own arrival time to ones governed by appointment books in which there is a limited number of slots to be filled. All these developments have resulted in multi-server appointment schedules becoming more prevalent.

When considering multi-server systems, there are different variants of appointment book design. First, we make the distinction between multiple, parallel single-server systems and pooled systems with one arrival process, but

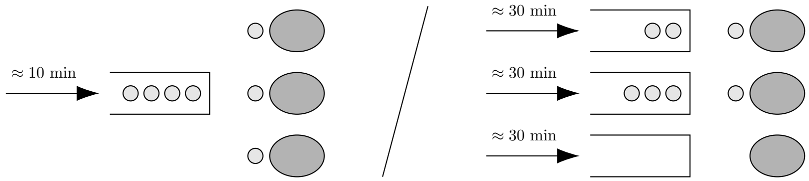


Figure 4.1: A comparison of a pooled system to parallel single-server systems

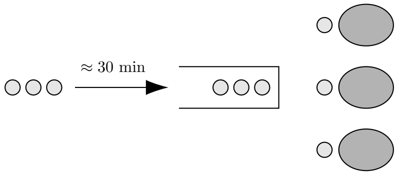


Figure 4.2: A system under batch-pooling

multiple servers. This latter system we term *pure pooling*. A representation of this comparison is given in Figure 4.1 where customers arrive approximately once every 10 minutes to the pooled system, and once every 30 minutes to each of the single parallel servers.

In this pure pooled case, however, there is the disadvantage that each individual server's appointment book cannot be distilled in advance with any confidence. Moreover, it is reported that in practice a single-queue structure in such a multi-server case has a detrimental effect on the average service times (Song et al. 2015, Shunko et al. 2018). However, by having customers arrive in *batches*, e.g., 3 customers arriving every 30 minutes as in Figure 4.2, we may be able to derive many of the benefits of pooling while still enjoying the managerial advantages of having a unique appointment book per server, such as synchronized appointments and unique appointment books per server. The appointment book will not be as certain or detailed as in a single-server system, but it will be more regular than in a multi-server queue with singular arrivals. Furthermore, a significant reduction in expected waiting or idle times from pooling may still be achieved. Queues with batch arrivals may thus strike a desirable balance between the two extremes of the single server queue and the multi-server queue with singular arrivals.

The rest of the paper proceeds as follows. In Section 4.2, we summarize relevant literature. In Section 4.3, we formally describe the problem, develop our model, extend it to include no-shows, and establish convexity of the objective function. Using the model, we run a comprehensive selection of experiments in Section 4.4. Finally, in Section 4.5, we provide managerial insights and conclude. Throughout this paper we will use the term *customer* to refer to those served by the queue, unless we are referring to an explicit healthcare setting.

4.2 Literature

This paper specifically addresses the theoretical component of constructing a $D^b/M/c$ system with deterministic batches of size b which can grant insight and serve as a point of comparison for further numerical study. We employ similar techniques to those used in Kendall (1953) and Tijms (2003) and we follow the notational conventions of the latter. In the remainder of this review, we consider the field of appointment scheduling literature through the lenses of batch and multi-server settings and will consider works that have a methodological resemblance.

4.2.1 Single Server Systems with Singular Arrivals

The single server appointment scheduling problem that has centered on the minimization of idle and waiting times has been extensively studied, resulting in a wide and dense field of available approaches Ahmadi-Javid et al. (2017), ever since the first analytical study of the D/M/1 queue in Jansson (1966). Relevant convexity properties for optimization of the single server setting have been proven, see e.g., Theorem 1 in Kuiper et al. (2023), in which the expected waiting and idle times are shown to be convex in the inter-arrival times — for the discrete equivalent of the appointment scheduling problem, see Theorem 3 in Zacharias and Yunes (2020).

In this paper we consider the system in stationarity, see for example Kuiper et al. (2017). This is not a strong assumption, as can be seen in Hassin and Mendel (2008) where it is found that restricting a schedule to be equally spaced has little negative effect on the cost of the schedule and echoes the statement of Stein and Côté (1994) that equally spaced inter-arrivals are a realistic restriction to the scheduling problem. Additionally, Kuiper et al. (2015) show that the plateau of the dome-shaped solution rapidly approaches the steady state solution as the number of customers increases.

4.2.2 Parallel Server Systems with Singular Arrivals

Multi-server systems in which servers operate in parallel are generally intractable as the Lindley recursion fails to apply. As a consequence keeping track of the system is more complicated, see Kiefer and Wolfowitz (1955), and thus waiting times become intractable. As pointed out by Grassmann (1988), there are different frameworks available to analyze these types of queues with reasonable accuracy. The presented approaches, however, do not adequately cover the framework of appointment scheduling in which the arrival moments of customers, or even batches of customers, have to be optimized.

Various studies have centered on settings with multiple resources in parallel, see Denton et al. (2010b) for a two-stage study. El-Sharo et al. (2015) considers an overbooking model in which each server has its own appointment schedule, but overbooked customers or customers not seen in their allotted slot will be 'shared' and can thus be seen by any server. Discrete event simulation is a popular method to study multiple server systems with parallel resources see, among others, Vanden Bosch and Dietz (2000), Sickinger and Kolisch (2009), Sun et al. (2011), Rohleder and Klassen (2002).

A notable attempt of deriving an optimization framework for multi-server appointment scheduling is found in Zacharias and Pinedo (2017), wherein they rule out service-time variability and consider no-shows as the single source of variation; under some settings the resulting schedules indeed feature a batch structure. Another approach, which encompasses both sources of variation, is to use realized schedules and apply machine learning to derive a load-based appointment scheduling heuristic that renders near optimal solutions for the multi-server case as done in Soltani et al. (2019). In Chapter 3 of this dissertation, also published as Kuiper and Lee (2022), a complete methodology for both service-time variation and no-shows is provided that relies on phase-type distributed service times such that the queue and status of the system can be tracked at all times, allowing evaluation and optimisation of an objective function. Furthermore, they show that solutions quickly converge to a steady-state, which is nearly equivalent to schedules obtained by considering a heavy-traffic regime; echoing the finding for the single-server case in Kuiper et al. (2017).

Analyzing a queueing system in stationarity has resulted in a considerable stream of works. In particular Kendall (1953) investigated $GI/M/c$ queues by means of their embedded Markov chains, using a geometric tail approach to solve the problem of an infinite state space. We extend Kendall's (1953) work to accommodate batch arrivals in a deterministic, appointment scheduling setting.

4.2.3 Server Systems with Batch Arrivals

In the creation of an appointment schedule, batches may also be of a determined size. In fact, the eponymous 'Rule of Bailey' derived from Bailey (1952) stipulates beginning an appointment schedule with a batch size of two. Not long after the works of Bailey (1952) and Welch and Bailey (1952), Soriano (1966) compared two scheduling systems: singular arrivals versus batches of two patients arriving at a time, relying on the fact that serving two patients can be modelled as two stages of a single service time. In Fries and Marathe (1981), variable-sized multiple batch appointment system are studied by means of a dynamic programming approach assuming a single server with exponentially distributed service times, in which the arrivals of batches are equally spaced. Recently, these types of systems have received attention by Srinivas and Choi (2022). Note that the global optimality of the solution in these system is guaranteed by the results of Zacharias and Yunes (2020) as long as the intervals are equally sized, which has been the case in the aforementioned works.

There is also a wide field of analytical work that accounts for batch arrivals in a general multi-server model. For example Liu and Liu (1998) run experiments varying the number of servers in case of different block appointment schedules to eventually come to a sub-optimal heuristic. Most analytical research into batch arrivals has in fact looked at batches of random size. This is understandable as in most applications of queueing theory the arrivals process is considered to be beyond the practitioner's control. For example, Zhao (1994) uses generating functions to investigate the $GI^X/M/c$ model, a queue with generally distributed arrivals of batches of varying size X, with multiple servers that have memoryless service times. Laxmi and Gupta (2000) and Chaudhry and Kim (2016) extend this model to additionally include a buffer of finite size. Rather than generating functions, they derive the transition probabilities explicitly and conduct a series of experiments to study the loss-probability of the system. They do not consider idle times, nor give the explicit form of the waiting time distribution, nor consider how the queue may be used in optimized appointment schedules. Gontijo et al. (2011) extend the framework beyond simple arrival distributions by approaching the inter-arrival distribution with kernel-density estimates and apply it to a call center setting. All the works focus on finding elegant methods for characterising batch queues rather than investigating optimal appointment schedules and so they also do not consider additional questions pertaining to appointment scheduling.

4.2.4 Contribution

For a commonly chosen objective function we develop a tractable appointment scheduling model for multi-servers and batch arrivals, i.e., we study $D^b/M/c$ in the three directions: b the size of arriving batches, c the number of servers, and x the inter-arrival time between two subsequent batches. Employing this model we are able to analyse various and relevant appointment book design choices, such as varying the batch size and number of servers on an (optimal) appointment schedule, while also being able to account for the environmental factors of no-shows and walk-ins. This provides practitioners a tool with which to study with great detail the different design choices available in a multi-server appointment scheduling framework. We pay particular attention to the case of multiple servers and singular arrivals (*pure pooling*), and the case where the number of servers and the batch size are matched (*batch pooling*). Comparing these settings and contrasting it to the baseline of a set-up with a parallel number of single-server queues, we find that batch pooling reaps nearly all of the benefits of pooling, while having a desirable structure for implementation in practice.

4.3 Problem and Model Formulation

4.3.1 Preliminaries

In a system with singular arrivals we seek to minimize the objective function

$$\mathcal{F}(x) = \mathbb{E}[W(x)] + \omega\mathbb{E}[I(x)],$$

Where $\mathbb{E}[W(x)]$ and $\mathbb{E}[I(x)]$ are the expected long run waiting and idle teams respectively in some inter-arrival time x. Note that in a system with batch arrivals each batch member will experience their own expected waiting time. Denote batch members by $m = 1, \ldots, b$, let $\mathbb{E}[W^m(x)]$ be batch member m's expected waiting time, and denote by $\overline{W} := \frac{1}{b}\sum_{m=1}^{b}\mathbb{E}[W^m(x)]$ the mean average long run expected waiting time for a batch; $\overline{W}$ is now a cost in terms of each individual customer. Also considering idle time in terms of cost generated per customer and writing this as $\overline{I}$ for consistency we write the optimization problem for batch arrivals as:

$$\min_x \mathcal{F}(x) = \overline{W} + \omega\overline{I}. \tag{4.1}$$

Throughout the rest of this paper we omit the argument x.

In this paper we extend the work of Kendall (1953) by considering a queue with deterministic arrivals, multiple memoryless servers and, in our case, arrivals in batches of size b. In the remainder of this section we shall first revisit Kendall's model for the $D/M/c$ queue. We then extend this approach in Section 4.3.2, where we derive the balance equations for the steady state as well as the waiting time distributions, idle times and other performance metrics. In Section 4.3.3 we include no-shows and walk-ins. Lastly, in Section 4.3.4 we demonstrate that the objective function is convex in the inter-arrival time.

The Embedded Markov chain

Before considering $D^b/M/c$ queues in depth, it will be convenient to recount Kendall's approach (Kendall 1953) to analysing $D/M/c$ queues. Let π_k be the steady-state probability that an arriving customer observes k customers present in the system. The probabilities $\pi_0, \pi_1, \pi_2, \ldots$ are the limiting distribution of an embedded Markov chain, i.e. embedded between two customer arrivals. If a customer arrives to find k other customers in the system, then the system is in state k, so that the number of customers served until the next arrival in x units of time ranges from 0 to $k+1$. If 0 customers are served in this period of time, then we increment the state by one, from k to $k+1$; if $k+1$ customers are served the system will be empty upon the arrival of the next customer. Let p_{kj} be the probability of starting in state k and ending in state j by the next arrival, i.e. of there being exactly $k+1-j$ departures in time x.

Kendall (1953) constructs the limiting distribution of the embedded Markov chain as (adapting the notation only slightly):

$$(\pi_0, \pi_1, \ldots, \pi_{c-2}, \pi_{c-1}, \eta\pi_{c-1}, \eta^2\pi_{c-1}, \ldots), \tag{4.2}$$

where $\eta, \eta^2, \ldots$ form the geometric tail, a discussion on which can be found in Tijms (2003, p. 111) and whose derivation will be given later in this chapter in Section 4.3.1. This geometric tail allows the balance equations to be written as a finite system of linear equations:

$$\begin{aligned}\pi_j &= \sum_{k=j-1}^{c-2} \pi_k p_{kj} + \pi_{c-1} p^*_{c-1,j}, \quad 1 \leq j \leq c-1 \\ \sum_{j=0}^{c-2} \pi_j &+ \frac{\pi_{c-1}}{1-\eta} = 1. \end{aligned} \tag{4.3}$$

Transition Probabilities

Let F_A be the cumulative distribution function of the inter-arrival random-variable. Throughout this paper we will suppose deterministic inter-arrival duration, but in some cases we will use the notation F_A to indicate where the inter-arrival process comes into play.

Case I: We consider the p_{kj} from display (4.3) for $k \leq c-1$, $j \leq k+1$. We go from state k to state j if there are $k+1-j$ departures in time x. The probability of a single departure in time x is given by $1 - e^{-\mu x}$. By virtue of deterministic inter-arrivals, the total number of departures in this time period is binomially distributed:

$$\begin{aligned} p_{kj} &= \int_0^\infty \binom{k+1}{j} e^{-\mu t j} (1 - e^{-\mu t})^{k+1-j} dF_A(t) \\ &\overset{det.}{=} \binom{k+1}{j} e^{-\mu x j} (1 - e^{-\mu x})^{k+1-j}. \end{aligned} \tag{4.4}$$

Case II: We now turn our attention to the $p^*_{c-1,j}$. First consider that if all servers are occupied so that the next arrival must join the queue we first need to service $k+1-c$ customers until only c remain and then we continue as in display (4.4). Note that the time to service $k+1-c$ customers is Erlang$_{k+1-c}$ distributed; and in the time remaining a further $c-j$ customers must be served, this yields:

$$\begin{aligned} p_{kj} = \int_0^\infty \int_0^t & \binom{c}{j} e^{-\mu(t-u)j} (1 - e^{-\mu(t-u)})^{c-j} \\ & \cdot (c\mu)^{k+1-c} \frac{u^{k-c}}{(k-c)!} e^{-c\mu u} \, du \, dF_A(t) \\ \overset{det.}{=} & \binom{c}{j} e^{-j\mu x} c\mu \int_0^x \frac{(c\mu u)^{k-c}}{(k-c)!} (e^{-\mu u} - e^{-\mu x})^{c-j} \, du, \end{aligned} \tag{4.5}$$

$$\text{for } 0 \leq j \leq c \leq k.$$

Writing out the expression for the $p^*_{c-1,j}$ from (4.3), we find the following, where $p_{c-1,j}$ is as in display (4.4), and the p_{kj}, $k \geq c$ are as in display (4.5):

$$\begin{aligned} p^*_{c-1,j} &= p_{c-1,j} + \eta p_{c,j} + \eta^2 p_{c+1,j} + \dots \\ &= p_{c-1,j} + \sum_{\ell=0}^{\infty} \eta^{\ell+1} \binom{c}{j} e^{-j\mu x} c\mu \int_0^x \frac{(c\mu t)^\ell}{\ell!} (e^{-\mu t} - e^{-\mu x}) dt \\ &= p_{c-1,j} + c\mu\eta \binom{c}{j} e^{-j\mu x} \int_0^x e^{c\mu\eta t} \left(e^{-\mu t} - e^{-\mu x}\right)^{c-j} dt. \end{aligned} \tag{4.6}$$

Geometric Tail

To find η we use the one-step transitions $\pi_j = \sum_{k=0}^{\infty} \pi_k p_{kj}$ and consider $j \geq c$, in particular and without loss of generality $j = c$.

$$\begin{aligned} \pi_c &= \sum_{k=c-1}^{\infty} \pi_k p_{kc} \\ &= \pi_{c-1} p_{c-1,c} + \pi_c p_{c,c} + \pi_{c+1} p_{c+1,c} + \dots \\ \Rightarrow \eta &= p_{c-1,c} + \eta p_{c,c} + \eta^2 p_{c+1,c} + \dots \end{aligned} \tag{4.7}$$

Note $p_{c-1,c}$ is the probability of no departures between two arrivals, $p_{c,c}$ is the probability of one departure, and so on. Also note that there are always enough customers in the queue to replenish servers which complete service and so these p are Poisson distributed. Letting the duration of inter-arrivals A have distribution F_A, we have

$$\begin{aligned} \eta &= \sum_{\ell=0}^{\infty} \eta^\ell \int_0^\infty \frac{(c\mu t)^\ell}{\ell!} e^{-c\mu t} dF_A(t) \\ &\overset{(*)}{=} \int_0^\infty \sum_{\ell=0}^{\infty} \eta^\ell \frac{(c\mu t)^\ell}{\ell!} e^{-c\mu t} dF_A(t) \\ &= \int_0^\infty e^{-(1-\eta)c\mu t} dF_A(t) \\ &\overset{det.}{=} e^{-(1-\eta)c\mu x}. \end{aligned} \tag{4.8}$$

Where step $(*)$ may be performed when $(1-\eta)c\mu t > 0$, which is when $\eta = \int_0^\infty e^{-(1-\eta)c\mu t} dF_A(t)$ has a root in $(0, 1)$.

Waiting Times

Here we report the waiting time distributions for the customers in Kendall's model. We again refer to Tijms (2003, p. 400). A customer who waits must wait until $k+1-c$ others have been served. This yields:

$$1 - W(t) = \sum_{k=c}^{\infty} \pi_j \sum_{j=0}^{k-c} e^{-c\mu t} \frac{(c\mu t)^j}{j!} = \frac{\eta}{1-\eta} \pi_{c-1} e^{-c\mu(1-\eta)t},$$

where $\sum_{j=0}^{k-c} e^{-c\mu t} \frac{(c\mu t)^k}{k!}$ is the probability that fewer than $k+1-c$ customers are served in time t. It is easy to see that the probability that the steady-state customer must wait is $\frac{\eta}{1-\eta}\pi_{c-1}$. The expectation is found quite straightforwardly as

$$\mathbb{E}[W] = \int_0^{\infty} 1 - W(t) dt = \frac{\eta \pi_{c-1}}{c\mu(1-\eta)^2}. \tag{4.9}$$

4.3.2 The Extension to Batches

Balance Equations

We extend the geometric tail approach of Kendall (1953). We will construct a finite system of equations by beginning the geometric tail from some M; Kendall chooses $M = c-1$. As in our case satisfactory values of M must be found by experimentation, we will consider this value only in general terms during the following derivations. We thus amend (4.2) to be of the form

$$(\pi_0, \pi_1, \ldots, \pi_{M-1}, \pi_M, \eta\pi_M, \eta^2\pi_M, \ldots). \tag{4.10}$$

Under $D^b/M/c$, when a batch of size b arrives to find k customers already in the system we increment the state from k to $k+b$, and then drain customers until the next arrival. Let $(\alpha)^+ := \max\{\alpha, 0\}$ be the positive part of argument α. The general form of the balance equations for both single arrivals and batches is given by

$$\begin{aligned} \pi_j &= \sum_{k=(j-b)^+}^{M-1} \pi_k p_{kj} + \pi_M p^*_{Mj}, \quad 1 \le j \le M \\ &\sum_{j=0}^{M-1} \pi_j + \frac{\pi_M}{1-\eta} = 1. \end{aligned} \tag{4.11}$$

As the value of M is uncertain (indeed it is free to be chosen by the practitioner) we must not only amend the probabilities from Section 4.3.1, but we must also consider additional cases as we must explicitly consider the one-step transition probabilities for a saturated system ($k > c$) in the balance equations.

Transition Probabilities

There are five possible cases that we must consider, these are summarized in Figure 4.3. We first consider $j \le c$, in which case we must differentiate $k+b \le c$

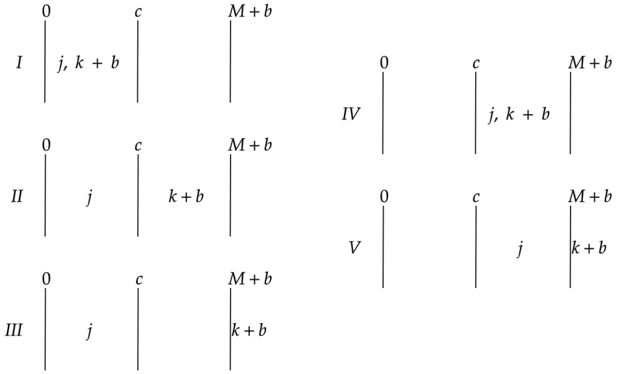


Figure 4.3: The five possible cases in which a different probability distribution is required.

(case I), $c < k+b < M+b$ (case II), and $k+b = M+b$ (case III). We then consider $j > c$, in which case we must differentiate $k+b < M+b$ (case IV), and $k+b = M+b$ (case V). We explicitly write $k+b$ to emphasize that k is the state of the Markov chain immediately prior to an arrival.

Case I: $j \leq c$, $k+b \leq c$. That is after the arrival of a batch, we have c or fewer customers in the system. We term the resulting distribution *Binomial.*

$$p_{kj} = \binom{k+b}{j} e^{-\mu x j}(1-e^{-\mu x})^{k+b-j}. \tag{4.12}$$

Case II: $j \leq c$, $c < k+b < M+b$. Now after the arrival of a batch we have more than c customers in the system, and we wish to know the probability that we drain down to some $j \leq c$. We must first service $k+b-c$ customers, and then service the remaining $c-j$. Proceeding as in equation (4.5) we find the following, terming this distribution *Erlang-Conditioned Binomial*, or *ECB.*

$$p_{kj} = \binom{c}{j} e^{-j\mu x} c\mu \int_0^x \frac{(c\mu u)^{k+b-c-1}}{(k+b-c-1)!}(e^{-\mu u} - e^{-\mu x})^{c-j}\, du, \qquad 0 \leq j \leq c \leq k. \tag{4.13}$$

Case III: $j \leq c$, $k+b = M+b$. The boundary case for the embedded Markov chain is found when $k = M$. As this case captures the infinite state-space, we term the resulting distribution *Infinite ECB.*

$$\begin{aligned} p^*_{Mj} &= \sum_{\ell=0}^{\infty} \eta^\ell p_{M+\ell,j} \\ &= \sum_{\ell=0}^{\infty} \eta^\ell \binom{c}{j} e^{-j\mu x} c\mu \int_0^x \frac{(c\mu u)^{M+\ell+b-c-1}}{(M+\ell+b-c-1)!}(e^{-\mu u} - e^{-\mu x})^{c-j} du \end{aligned}$$

$$= \binom{c}{j} e^{-j\mu x} c\mu \eta^{c-M-b+1}$$
$$\cdot \int_0^x \left(e^{c\mu u\eta} - \sum_{\ell=0}^{M+b-c-2} \frac{(c\mu u\eta)^\ell}{\ell!} \right) (e^{-\mu u} - e^{-\mu x})^{c-j} du. \tag{4.14}$$

Case IV: $j > c$, $k+b < M+b$. If $j > c$ we remain at all times in a system with all servers occupied and as such there are always customers in the system who can replenish servers. The time taken to service $k+b-j$ customers is thus Poisson distributed, and we term this distribution simply *Poisson*.

$$p_{kj} = e^{-c\mu x} \frac{(c\mu x)^{k+b-j}}{(k+b-j)!}. \tag{4.15}$$

Case V: $j > c$, $k+b = M+b$. We again find ourselves in the boundary case. This we term *Infinite Poisson*.

$$p^*_{Mj} = p_{Mj} + \sum_{\ell=1}^{\infty} \eta^\ell e^{-c\mu x} \frac{(c\mu x)^{M+\ell+b-j}}{(M+\ell+b-j)!}$$
$$= \eta^{j-M-b} e^{-c\mu x} \left[e^{c\mu x\eta} - \sum_{\ell=0}^{M+b-j-1} \frac{(c\mu x\eta)^\ell}{\ell!} \right]. \tag{4.16}$$

Geometric Tail

We find the geometric tail from expression (4.10):

$$x_{M+b} = \sum_{k=M}^{\infty} x_k \int_0^\infty \frac{(c\mu t)^{k-M}}{(k-M)!} e^{-c\mu t} dF_A(t)$$
$$\Rightarrow \eta^b = \sum_{\ell=0}^{\infty} \eta^\ell \int_0^\infty \frac{(c\mu t)^\ell}{\ell!} e^{-c\mu t} dF_A(t)$$
$$\overset{det.}{\Rightarrow} \eta = e^{-(1-\eta)\frac{c\mu x}{b}},$$

as long as $c\mu x/b > 1$.

Performance Measures

For the purpose of optimisation we are interested in long run average waiting time and long run average idle time. The long run average waiting time requires the most derivation and discussion and is derived first.

When a batch of customers indexed $m = 1, 2, \ldots, b$ arrives (who are seen in order of their index) the first customer must wait if there are at least c servers occupied, the second if there are at least $c-1$, and so on, the general form being $c-m+1$ (of course if $c-m+1$ is negative then 0 is "at least"

$c-m+1$). In deriving the waiting times for the $D/M/c$ queue we made use of the fact that $\pi_j = \eta^{j-c+1}\pi_{c-1}$ for $j \geq c-1$, i.e., we made use of the geometric tail. In general, for batch arrivals we cannot be guaranteed that π_{c-m+1} is part of this geometric tail, prohibiting a simple expression. We can, however, still write this succinctly:

$$\begin{aligned}
1 - W^m(t) &= \sum_{j=0}^{\infty} \pi_j \sum_{k=0}^{j-c+m-1} e^{-c\mu t}\frac{(c\mu x)^k}{k!} \\
&= \sum_{j=0}^{M} \pi_j \sum_{k=0}^{j-c+m-1} e^{-c\mu t}\frac{(c\mu t)^k}{k!} \\
&\qquad\qquad + \pi_M \sum_{j=M}^{\infty} \eta^{j-M} \sum_{k=0}^{j-c+m-1} e^{-c\mu t}\frac{(c\mu t)^k}{k!}.
\end{aligned}$$

Where we let the contents of an empty sum equal 0. Integrating this expression from zero to infinity and rewriting for closed form we find:

$$\begin{aligned}
\mathbb{E}[W^m] = &\left[\sum_{j=0}^{M} \pi_j \frac{(j-c+m)^+}{c\mu}\right] \\
&\qquad + \frac{\eta\pi_M}{c\mu(1-\eta)^2}\left[(M+m-c)(1-\eta)+1\right]. \qquad (4.17)
\end{aligned}$$

We will consider the mean expected waiting time for a batch, $\overline{W} := \frac{1}{b}\sum_{m=1}^{b} \mathbb{E}[W^m]$, though one could for example consider the variables W^1 or W^b to investigate the waiting time distributions for priority customers or the worst-off customer respectively.

For optimisation we are also interested in idle times. In a $G/G/1$ queue with singular arrivals it is simple to calculate idle time as $x-1/\mu$ and utilization as $1/(x\mu)$. In a multi-server system with batch arrivals we find an idle time over n arrivals of $n(cx-b/\mu)$ which divided by the number of customers bn gives $\mathbb{E}[I] = \lim_{n\to\infty} \frac{n(cx-b/\mu)}{nb} = \frac{c}{b}x - \frac{1}{\mu}$. We also have utilization $u = \frac{b}{cx\mu}$ and throughput per unit of service time $\overline{T} = b/x$.

4.3.3 Including Environmental Factors: No-Shows and Walk-Ins

Let N be the number of no-shows from a batch of size b. N is then Binomial(b,q) distributed, where q is an individual arrival's no-show probability. Let G be the random variable representing batch size (one can imagine G as standing for *group*, we avoid B as this is often used for *busy period*). In the case of no-shows $G = b - N$. Let $U(t)$ be the number of walk-ins over a period of time t. This gives us $G(t) = b - N + U(t)$. It is reasonable to let $U(t)$

be Poisson distributed with some rate dependent upon t. In practice we will truncate the domain of $U(t)$ to $\{0,\dots,n\}$ where $P(U(t)=n+1)\le 10^{-6}$ as this lends $G(t)$ a finite domain. Lastly, let $\nu_\gamma = \mathbb{P}(G=\gamma)$ be the probability that a batch contains γ customers.

Amendments to the Probabilities

We continue to use the balance equations as in Display (4.11), but with slightly amended transition probabilities. To amend the transition probabilities to permit a random batch G on support $\gamma \in \{0,\dots,|G|\}$, we write for each p_{kj} from Section 4.3.2 a superscript γ such that in the deterministic case $p_{kj} := p^b_{kj}$. For random batches we then get

$$\begin{aligned} p_{kj} &= \sum_{\gamma=0}^{|G|} p^{\gamma}_{kj}, \text{ and} \\ p^*_{Mj} &= \sum_{\gamma=0}^{|G|} p^{*\gamma}_{Mj}. \end{aligned}$$

The geometric tail is now found as follows:

$$\begin{aligned} x_{M+b} &= \sum_{\gamma=0}^{b} \nu_\gamma \sum_{k=M+b-\gamma}^{\infty} x_k \int_0^\infty \frac{(c\mu t)^{k-M-b+\gamma}}{(k-M-b+\gamma)!} e^{-c\mu t} dF_A(t) \\ \Rightarrow \eta^b &= \sum_{\gamma=0}^{b} \nu_\gamma \eta^{b-\gamma} \sum_{\ell=0}^{\infty} \eta^\ell \int_0^\infty \frac{(c\mu t)^\ell}{\ell!} e^{-c\mu t} dF_A(t) \\ \overset{det.}{\Rightarrow} \eta^b &\Bigg/ \sum_{\gamma=0}^{b} \nu_\gamma \eta^{b-\gamma} = e^{-(1-\eta)c\mu x}. \end{aligned} \tag{4.18}$$

Performance Measures

Under no-shows and walk-ins, as we do not count the waiting time of people who do not turn up the expected long run waiting time becomes

$$\overline{W} = \sum_{\gamma=1}^{|G|} \nu_\gamma \sum_{m=1}^{\gamma} \mathbb{E}[W^m]/\gamma,$$

which is the sum of the expected long run average waiting times for each scenario γ, weighted by the probability of that scenario occurring. Expected idle time then becomes $\mathbb{E}[I] = \frac{c}{\mathbb{E}[G]}x - \frac{1}{\mu}$.

In our experiments we will also consider the detail of the expected waiting times per batch member. While ν_γ is the probability that γ customers arrive in a batch, let ξ_m be the probability that the batch that arrives contains at least

m customers. That is we let $\xi_0 = 1$ and $\xi_m = \xi_{m-1} - \nu_{m-1}$ for $m > 0$. Then with probability ξ_m batch member m arrives and experiences expected waiting time $\mathbb{E}[W^m]$. To give an honest reflection of the expected waiting times of those customers who do in fact arrive, we will report batch members' expected waiting times as $\xi_m \mathbb{E}[W^m]$.

4.3.4 Convexity of the Objective Function

In our optimisation we apply a variant of golden-section search. If the function that we are minimising is convex, we are guaranteed that a unique minimum exists and golden-section search will eventually converge on this minimum. We now turn to the question of whether our objective function is indeed convex in the inter-arrival time that we choose. Expected idle time $\mathbb{E}[I] = \frac{c}{b}x - \frac{1}{\mu}$ is clearly linear and thus convex in x and the objective function $\mathcal{F}$ is a linear combination of idle and waiting times. We must thus ask whether expected waiting time is also convex in x.

Let $x := A(\theta)$ be an arrival process depending upon parameter θ. Let $N_k(\theta)$ denote the number of customers in the system at arrival of the k-th batch and let $W_k^m(\theta)$ denote the waiting time of member m of the k-th batch. Furthermore, let $D(n, t)$ be the number of survivors at time t in a pure death process, starting with n survivors at time 0. Let the symbol $\overset{st}{=}$ denote stochastically equal.

We apply Theorem 6.2 of Shaked and Shanthikumar (1990), which states:

Lemma 4.1 (Th. 6.2 of S. & S.). *Suppose $\{A(\theta), \theta \in \Theta\} \in SDCV(sp)$, i.e., it is stochastically decreasing and concave in the sample path sense. If $\{N_0(\theta), \theta \in \Theta\} \in SICX(sp)$, i.e., stochastically increasing and convex in the sample path sense, then*

1. *$\{N_k(\theta), \theta \in \Theta\} \in SICX(sp)$, and*
2. *$\{W_k^m(\theta), \theta \in \Theta\} \in SICX(sp)$.*

This theorem first establishes convexity of $N_k(\theta)$ and extends this via closure properties to $W_k(\theta) \overset{st}{=} \sum_{i=1}^{[N_k(\theta)-c]^+} B_i$, with B_i exponentially distributed random variables with rate μc.

The above theorem applies Theorem 6.3 of Shaked and Shanthikumar (1990) as a lemma to establish convexity of the pure death process, and thus $N_k(\theta)$ (the three expressions given at the end of Theorem 6.3 being conditions for convexity when the initial conditions are satisfied). Let $\gamma(n)$ be the death rate (i.e., service rate) for a number of customers n.

Lemma 4.2 (Th. 6.3 of S. & S.). *Suppose $\gamma(n)$ is increasing and concave in n. Then for any choice of $(y_i, t_i), i = 1, 2, 3, 4$, such that $y_1 \le y_2 \le y_3 \le y_4$, $y_1 + y_4 = y_2 + y_3$, $t_1 \le \min\{t_2, t_3\}$, $t_4 \ge \max\{t_2, t_3\}$, and $t_1 + t_4 = t_2 + t_3$, there exist four random variables $\hat{X}_i, i = 1, 2, 3, 4$, defined on a common probability* *space such that*

$$\begin{aligned} \hat{X}_i &\overset{st}{=} D(y_i, t_i), && i = 1, 2, 3, 4, \\ \max\{\hat{X}_1, \hat{X}_2, \hat{X}_3\} &\leq \hat{X}_4, && \textit{almost surely, and} \\ \hat{X}_1 + \hat{X}_4 &\geq \hat{X}_2 + \hat{X}_3 && \textit{almost surely.} \end{aligned}$$

The appropriate choice of $\gamma(n)$ is $\gamma(n) := \min\{n, c\}\mu$. We now formalize our theorem and proof:

Theorem 4.1. *$N_k(\theta)$ and $W_k^m(\theta)$ are stochastically increasing and convex in the sample path sense in the inter-arrival time $x(\theta) := A(\theta)$.*

Proof. In Lemma 4.1 we choose $x := A(\theta) = \hat{\theta} - \theta$, which is decreasing and linear and therefore concave, and let $\hat{\theta} \to \infty$. We draw your attention to the fact that the (in)equalities in Lemma 4.2 are unchanged by the addition of a batch size b to each y_i. Note lastly that when we start with an empty system $N_0(\theta) = 0$ for all values of θ, satisfying the initial condition of Lemma 4.1 and thus proving the desired convexity properties for $N_k(\theta)$. Consider now that batch member m's waiting time is given by

$$W_k^m(\theta) \overset{st}{=} \sum_{i=1}^{[N_k(\theta)+b-1-c]^+} B_i,$$

which adheres to the closure properties, thereby demonstrating the desired convexity properties for batch member m's waiting time. □

Remark 4.1. *Sample path convexity applies for the* random variables $N_k(\theta)$ *and* $W_k^m(\theta)$ *and thus also applies for their expectations* $\mathbb{E}[N_k(\theta)]$ *and* $\mathbb{E}[W_k^m(\theta)]$. *Let $k \to \infty$ for the long run expected waiting time.*

Remark 4.2. *This proof holds for no-shows, as by the sample-path argument b may stand in for any draw from the distribution G. It is however as yet unclear how convexity holds for walk-ins as both the inter-arrival time and the size of the batch depend upon the same time parameter.*

Remark 4.3. *We have shown convexity of batch k's waiting time in the inter-arrival times preceding their arrival. This extends to the steady state and answers an open question given in Chapter 3 of this dissertation. This does not, however, translate to the transient case where one must find an optimal vector $(x_k)_{k=1}^n$.*

4.4 Results

4.4.1 A Trade-Off between Utilization and Waiting Time

In Figure 4.4 we compare the mean average waiting times for 5 systems for different values of utilization. From top to bottom we graph the waiting times for: a single server with singular arrivals; 2 servers with batches of 2 (batch pooling); 2 servers with singular arrivals (pure pooling); 4 servers with batches of 4 (batch pooling); and 4 servers with batches of 1 (pure pooling). We see that mean average waiting times increase far steeper for the singular system than for all other systems. There also appears to be little distinction between batch and pure pooling for multiple servers, at least when compared with a single server.

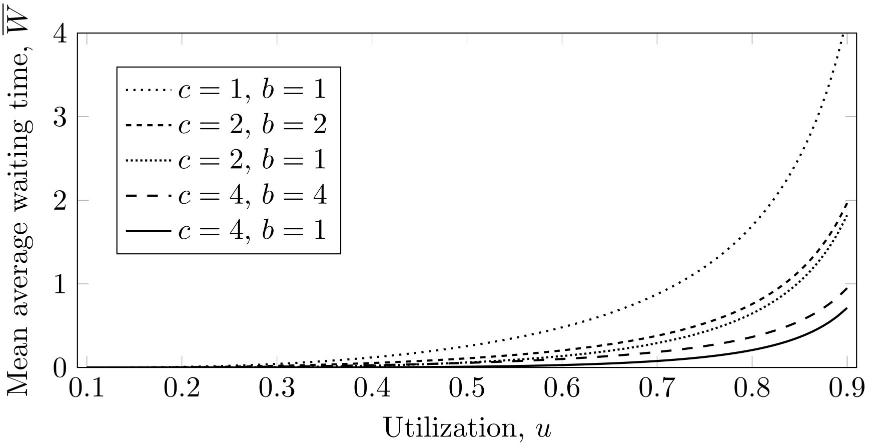


Figure 4.4: Mean average waiting times for a fixed 90% utilization, comparing a singular system to pure and batch pooling.

In Figure 4.5, we compare these same five systems with one another when optimized for different costs of idle time in equation (4.1). From top to bottom, we graph the waiting times (increasing) and idle times (decreasing) in the same order as reported in Figure 4.4. Both pure and batch pooling outperform a singular system in both idle and waiting times for all values of the cost of idle time. Batch pooling always has a greater waiting time than pure pooling for the same number of servers and a slightly higher idle time.

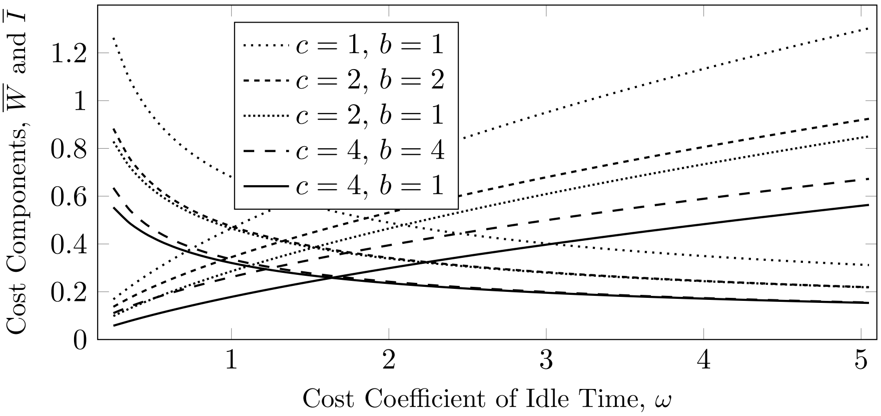


Figure 4.5: Mean average waiting time (*increasing*) and idle time (*decreasing*) in an optimised system for different values of ω. Comparing a singular system to pure and batch pooling for $c = 2$ and $c = 4$.

		c						
		1	2	4	8	16	32	64
b	1	***4.1787***	***1.9330***	***0.8612***	**0.3629**	**0.1404**	**0.0475**	**0.0129**
	2	4.4790	***2.0143***	0.8859	0.3703	0.1426	0.0480	0.0130
	4	5.1957	2.3054	***0.9828***	0.4003	0.1512	0.0503	0.0134
	8	6.8118	3.0534	1.2951	**0.5131**	0.1861	0.0595	0.0154
	16	10.3150	4.7552	2.0800	0.8479	**0.3120**	0.0980	0.0243
	32	17.7080	8.4133	3.8447	1.6616	0.6628	**0.2321**	0.0642
	64	33.0242	16.0444	7.6044	3.4668	1.4942	0.5914	**0.2020**

Table 4.1: Mean expected long-run waiting time with 90% utilization.

4.4.2 Configuring the Appointment Book: Varying Batch Size and Number of Servers

Table 4.1 reports mean expected long run waiting times for varying batch sizes and number of servers for a fixed utilization of 90%. The bold printed values in Table 4.1 are combinations of batch size and number of servers of particular interest, namely all cases with $b = 1$ (pure pooling) and all cases where $b = c$ (batch pooling). The italic printed values correspond to those from Figure 4.4 when utilization is 90%. Fixing utilization fixes idle time per customer, such that 32 parallel servers will have the same throughput and the same accumulated idle time as a single system of 32 servers, holding batch size equal. We see in this table how singular systems are heavily outperformed in waiting time by any pooled system. We also see how for smaller values of c the difference in long run expected waiting times between pure and batch pooling are small. Also note that expected waiting times are lowest when $b = 1$ and c

is as large as possible. Therefore, optimization of our objective function in b and c is trivial.

In Tables 4.2 through 4.5 we consider performance measures at optimality, given costs of idle time of $\omega = 1$ and $\omega = 5$, for a single server system with singular arrivals, the single server system with dual arrivals that Soriano (1966) examines, and pure and batch pooling for 2, 4, and 8 servers. In Tables 4.3 and 4.5 we report for each choice of ω for each member of a batch what his or her expected waiting time would be.

In both Tables 4.2 and 4.4, pure pooling enjoys the lowest mean average waiting and idle times (and thus highest throughput and utilization), but these values are closely followed by those for batch pooling. The similarity in utilization between the pure and batch pooling settings in Table 4.4 are remarkable. In all examples both the batch and the pure pooling examples significantly outperform the single server case.

In Tables 4.3 and 4.5 we see how the expected waiting time per batch member grows as the batch member index grows. This is under the implicit assumption that all batch members arrive simultaneously, and in practice the observed waiting times may be lower. The waiting time of the lowest indexed (i.e. first) batch member is lower even than that of a customer in the equivalent pure pooled setting. This has managerial implications for systems with triage, where customers are assigned a position in the batch upon arrival, according to the severity of their situation. Table 4.3 also shows that the expected waiting times of all but the last batch members are still superior to those of a customer in the single server system. For the higher cost coefficient $\omega = 5$ in Table 4.5 this even holds for the last batch member. This demonstrates how batch pooling can capture the benefit of pure pooling in terms of utilization (and thus related metrics), while also still performing better from the customer's perspective than the singular system.

To compare with Soriano (1966), we also consider in Tables 4.2 and 4.4 the case where $c = 1$ and $b = 2$, i.e. dual arrivals to a single server. Corresponding to what was found by Soriano, $c = 1$, $b = 2$ benefits the server in terms of idle time (and slightly increases throughput), but at significant expense for customer waiting time; the second customer in the batch of course having to wait for the first customer to be served, as can be seen in Tables 4.3 and 4.5.

4.4.3 Designing Appointment Books in the Case of No-Shows or Walk-Ins

No-Shows

No-shows are generally detrimental to an appointment schedule, as they introduce variance into the system. Remarkably, batches work to mitigate the negative effects of no-shows when the cost placed on idle time, ω is low, that is $\omega \leq 1$. This is summarized in Table 4.6, where we see that for $c = 1, b = 2$ and $c = 8, b = 8$ the cost $\mathcal{F}_q$ is decreasing in all q. For all systems the cost decreases

c	b	x^*	$\overline{W}$	$\overline{I}$	η	$\overline{T}$	u
1	1	1.6803	0.4660	0.6803	0.3179	0.5952[a]	0.5952
1	2	3.2985	0.8681	0.6492	0.3327	1.2127	0.6063
2	1	0.7325	0.2858	0.4650	0.4407	1.3652	0.6826
2	2	1.4738	0.3448	0.4738	0.4347	1.3570	0.6785
4	1	0.3301	0.1785	0.3205	0.5574	3.0291	0.7573
4	4	1.3360	0.2617	0.3359	0.5433	2.9943	0.7486
8	1	0.1528	0.1144	0.2222	0.6599	6.5453	0.8182
8	8	1.2444	0.2157	0.2444	0.6348	6.4289	0.8036

Table 4.2: Equivalent systems, $\omega = 1$. [a] Throughput for each single server in isolation; multiply [a] by c to receive throughput for a system of c parallel servers. All other throughputs are reported for the entire system of servers.

$\mathbb{E}[W^m]$		Batch member, m							
		1	2	3	4	5	6	7	8
Servers, c	1	0.3681	1.3681	·	·	·	·	·	·
	2	0.2017	0.4879	·	·	·	·	·	·
	4	0.0840	0.1579	0.2963	0.5084	·	·	·	·
	8	0.0316	0.0500	0.0800	0.1279	0.1993	0.2951	0.4094	0.5324

Table 4.3: Expected waiting times for each batch member, $b = 2$, 4, or 8; $\omega = 1$.

c	b	x^*	$\overline{W}$	$\overline{I}$	η	$\overline{T}$	u
1	1	1.3133	1.2949	0.3133	0.5643	0.7615[a]	0.7615
1	2	2.6184	1.6446	0.3092	0.5681	1.5276	0.7638
2	1	0.6097	0.8449	0.2194	0.6632	1.6402	0.8201
2	2	1.2201	0.9187	0.2201	0.6623	1.6392	0.8196
4	1	0.2885	0.5599	0.1540	0.7456	3.4662	0.8666
4	4	1.1556	0.6688	0.1556	0.7434	3.4614	0.8654
8	1	0.1385	0.3764	0.1083	0.8111	7.2182	0.9023
8	8	1.1107	0.5154	0.1107	0.8075	7.2026	0.9003

Table 4.4: Equivalent systems, $\omega = 5$. [a] Throughput for a each single server in isolation; multiply [a] by c to receive throughput for a system of c parallel servers. All other throughputs are reported for the entire system of servers.

$\mathbb{E}[W^m]$		Batch member, m							
		1	2	3	4	5	6	7	8
Servers, c	1	1.1446	2.1446	·	·	·	·	·	·
	2	0.7261	1.1113	·	·	·	·	·	·
	4	0.4062	0.5493	0.7426	0.9771	·	·	·	·
	8	0.2253	0.2792	0.3471	0.4314	0.5321	0.6461	0.7677	0.8922

Table 4.5: Expected waiting times for each batch member, $b = 2$, 4, or 8; $\omega = 5$.

	$c=1$	$c=1$	$c=2$	$c=2$	$c=4$	$c=4$	$c=8$	$c=8$
	$b=1$	$b=2$	$b=1$	$b=2$	$b=1$	$b=4$	$b=1$	$b=8$
$\mathcal{F}_0$	1.1462	1.5173	0.7508	0.8186	0.4990	0.5975	0.3366	0.4601
$\mathcal{F}_{0.1}$	1.1643	**1.4853**	0.7624	0.8368	0.5069	0.5996	0.3420	**0.4500**
$\mathcal{F}_{0.2}$	1.1741	**1.4581**	0.7682	0.8527	0.5107	0.6044	0.3446	**0.4430**
$\mathcal{F}_{0.3}$	**1.1727**	**1.4332**	**0.7666**	0.8646	**0.5096**	0.6111	**0.3439**	**0.4392**

Table 4.6: Total cost of systems subject to various no-show rates q when $\omega = 1$. Bold entries are less than the entry in the cell above, that is the cost of these systems is decreasing in q.

		$q=0.1$			$q=0.2$			$q=0.3$		
c	b	x_{NS}	$\overline{W}$	$\overline{I}$	x_{NS}	$\overline{W}$	$\overline{I}$	x_{NS}	$\overline{W}$	$\overline{I}$
1	1	1.5135	0.4826	0.6817	1.3411	0.4977	0.6764	1.1643	0.5095	0.6633
1	2	3.0296	0.8022	0.6831	2.7365	0.7478	0.7103	2.4205	0.7043	0.7289
2	1	0.6600	0.2957	0.4667	0.5855	0.3044	0.4638	0.5095	0.3110	0.4556
2	2	1.3414	0.3463	0.4905	1.2028	0.3492	0.5035	1.0581	0.3531	0.5115
4	1	0.2975	0.1848	0.3220	0.2641	0.1903	0.3204	0.2302	0.1944	0.3152
4	4	1.2126	0.2523	0.3474	1.0870	0.2457	0.3587	0.9586	0.2417	0.3695
8	1	0.1376	0.1185	0.2235	0.1223	0.1221	0.2225	0.1067	0.1248	0.2191
8	8	1.1259	0.1990	0.2510	1.0059	0.1856	0.2574	0.8846	0.1755	0.2637

Table 4.7: The effect of various no-show rates, $\omega = 1$. ^{a} Throughput for a each single server in isolation; multiply ^{a} by c to receive throughput for a system of c parallel servers. All other throughputs are reported for the entire system of servers.

in the no-show rate once this rate becomes sufficiently large as frequent no-shows result in very few customers waiting and allow very close inter-arrivals, reducing idle time. We see this beginning for some systems from $q = 0.3$. The division of this cost over expected waiting and idle times is shown in Table 4.7.

In Table 4.7 we see how pure pooling generally preserves idle time as the rate of no-shows increases, while batch pooling preserves waiting times; indeed expected waiting times even improve as q increases. There is a caveat to this, however: the expected waiting time of the first customer to arrive worsens in no-shows no matter what, as can be seen by comparing Table 4.8 with Table 4.3. This is particularly pertinent in a system with priority customers, where the advice would be to separate these customers from systems with a high risk of no-shows whenever possible. Indeed, if we follow a special priority customer who we imagine arrives with certainty to the steady state system — i.e. we consider $\mathbb{E}[W^1]$ and not $\xi_1\mathbb{E}[W^1]$ — then we see that this customer in particular suffers; we illustrate with the case $c = b = 8$, where the uncorrected expected waiting time of the first batch member $\mathbb{E}[W^1]$ is 0.1490 (c.f. 0.0601 in Table 4.8 and 0.0316 in Table 4.3)

It is also worth noting that these (optimized) systems see little effect on throughput from no-shows as the optimal inter-arrival time is adjusted accordingly. Comparing Tables 4.2 and 4.7 confirms the rule as reported in Çayırlı et al. (2012) that the optimal inter-arrival time given no-shows can be well approximated as $x_{\text{NS}} \approx (1 - q)x^*$. This approximate rule was also seen for different values of ω not shown here.

$\xi_m\mathbb{E}[W^m]$		Batch member m							
		1	2	3	4	5	6	7	8
Servers c	1	0.4593	0.3046	·	·	·	·	·	·
	2	0.2747	0.7373	·	·	·	·	·	·
	4	0.1331	0.2086	0.2508	0.1455	·	·	·	·
	8	0.0601	0.0876	0.1277	0.1789	0.2202	0.2082	0.1265	0.0357

Table 4.8: Expected waiting times for each batch member, $b = 2$, 4, or 8; $\omega = 1$. We set $q = 0.3$ as this extreme example emphasises the effect on the first batch member.

Walk-Ins

We begin our foray into walk-ins by considering in Table 4.9 how pure and batch pooling compare to an entirely unoptimized system, that is one which observes exclusively walk-ins. Note also that when $c = 1$ both pure pooling and batch pooling are reduced to a single server system. This last system we model by use of an $M/M/c$ queue, where we choose the arrivals rate $\lambda := 0.9c$ so that utilization is constant at 90%. Here we see, predictably, that pure pooling outperforms both batch pooling and exclusively walk-ins; though as the number of servers increases, the appointment book becomes meaningless and very difficult to govern as the inter-arrivals become ever shorter. What stands out, however, is that under the implicit assumption that all customers in a batch arrive simultaneously there comes a point where the average waiting time of the system with exclusively walk-ins is less than that of the batch pooled system. This all suggests that for sufficiently large systems it may be attractive simply not to optimize.

$\overline{W}$	c						
	1	2	4	8	16	32	64
Pure Pooling	4.1787	1.9330	0.8612	0.3629	0.1404	0.0475	0.0129
Batch Pooling	4.1787	2.0143	0.9828	0.5131	0.3120	0.2321	0.2020
Walk-ins only	9.0000	4.2632	1.9694	0.8769	0.3696	0.1432	0.0485

Table 4.9: Comparison in $\overline{W}$ to a system of only walk-ins at 90% utilization.

In Table 4.10, we explore the effects of walk-ins on optimized systems. We imagine that for designing the appointment book the practitioner has a fixed number of servers to distribute however they wish, for example in parallel systems of single servers, with batch pooling, or with pure pooling. The walk-in rate per system is scaled to the number of servers, so that each parallel server would see a rate of λ walk-ins per time unit, but two servers in a joint system would together see a rate of 2λ. We let the number of walk-ins over an inter-arrival period t have distribution Poisson$(t\lambda c)$. An implicit assumption in our formulation is that walk-in customers are also subject to the system's appointment book, and such are only admitted at regular intervals and are not seen upon their arrival; we imagine scheduled customers taking priority over walk-in customers, receiving a lower batch member index.

		$\lambda = 0.1c$			$\lambda = 0.2c$			$\lambda = 0.3c$		
c	b	x_{WI}	$\overline{W}$	$\overline{I}$	x_{WI}	$\overline{W}$	$\overline{I}$	x_{WI}	$\overline{W}$	$\overline{I}$
1	1	2.0432	0.6211	0.6966	2.5577	0.8288	0.6921	3.2671	1.1424	0.6499
2	1	0.8743	0.3373	0.4883	1.0799	0.3942	0.5083	1.3943	0.4728	0.5183
2	2	1.7509	0.4302	0.4900	2.1304	0.5504	0.4939	2.6466	0.7396	0.4753
4	1	0.3861	0.2009	0.3377	0.4642	0.2232	0.3539	0.5805	0.2477	0.3686
4	4	1.5589	0.3188	0.3487	1.8568	0.4020	0.3540	2.2569	0.5339	0.3457
8	1	0.1760	0.1257	0.2341	0.2073	0.1365	0.2454	0.2520	0.1473	0.2561
8	8	1.4337	0.2612	0.2539	1.6819	0.3281	0.2586	2.0115	0.4293	0.2557

Table 4.10: The effect of various walk-in rates, $\omega = 1$. [a] Throughput for a each single server in isolation; multiply [a] by c to receive throughput for a system of c parallel servers. All other throughputs are reported for the entire system of servers.

In Table 4.10 we report the optimized inter-arrival times as well as the average long run expected waiting and idle times. We report these only for $\omega = 1$ as this is the case where the differences were most apparent. The main impact of walk-ins is via the optimized inter-arrival time x_{WI}. Walk-ins force that times must be extended in all cases as compared with Table 4.2. This brings with it a confounding effect, as the greater x_{WI} the higher the variance in walk-ins and so the greater the cost to the system. Therefore, while all systems suffer from walk-ins, batch pooled and parallel systems suffer more than pure pooled system. The shorter inter-arrival times of pure pooled systems also mean that these systems see the fraction of walk-in customers. Finally, there is a slight improvement in long run expected idle time for the batch pooled systems when $\lambda = 0.3c$.

4.4.4 How Short-Run Appointment Schedules Approach their Long Run Counterparts

In Chapter 3 an analytical model is developed to study the transient problem of multi-server appointment scheduling, i.e., for a finite number of customers an optimal appointment schedule is derived. Their schedules revolve the fully pooled system without batch arrivals ($b = 1$). One of their key results is that the transient solutions quickly converge to a plateau solution, which coincides with the steady-state solution.

The purpose of the comparison is two-fold. Firstly, using their framework we can verify our results. Secondly, with this comparison we can illustrate how the steady state model presented in this paper can serve as a convenient jumping off point for studying more complicated systems (e.g., batch arrivals) and computationally intensive settings (e.g., many servers and many customers) in the steady state.

We can tailor the analytical model given in Chapter 3 to study batch scheduling in a transient setting. This can be done by restricting the problem such that in case of a batch size b, after an inter-arrival the next $b-1$ inter-arrival times are set to 0. In detail, where they study the problem (with c the number of servers) of finding the optimal schedule, the vector $\boldsymbol{x}$, that describes the inter-arrival times:

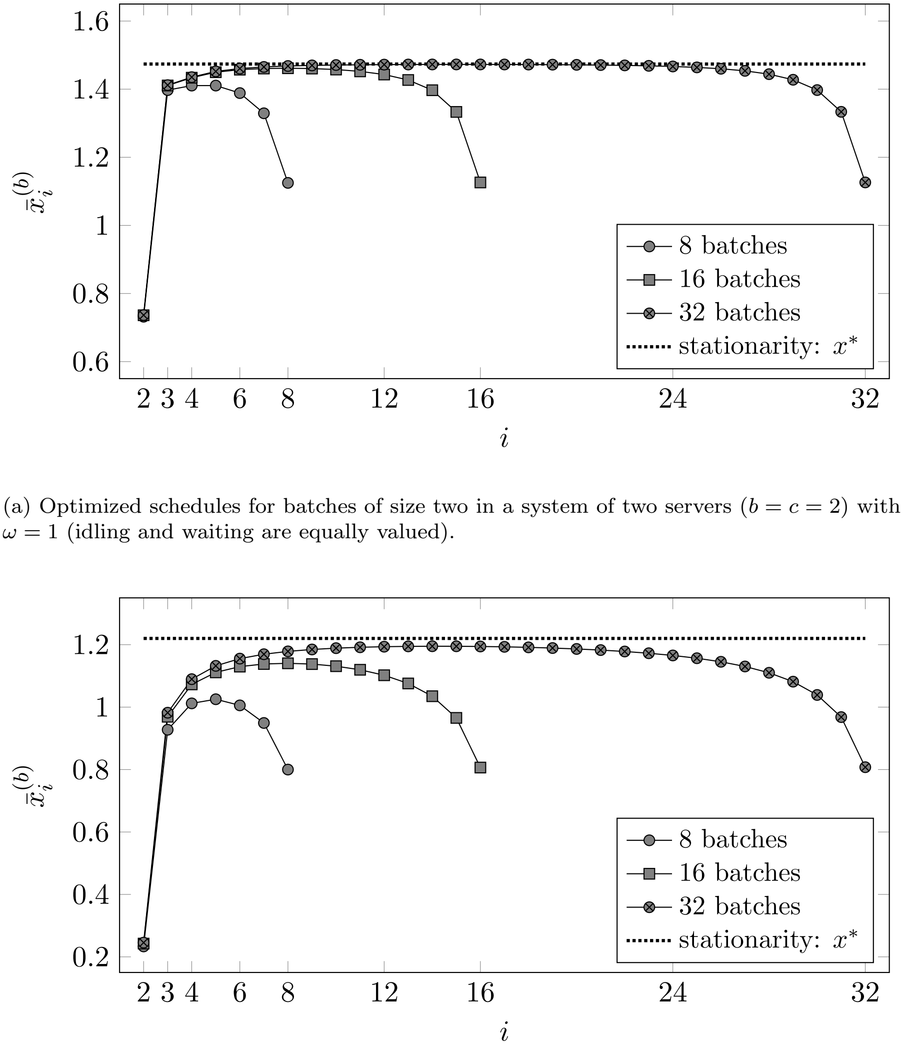
$$\begin{aligned}\boldsymbol{x} = \min_{\boldsymbol{x}} \mathcal{F}(\boldsymbol{x}) = \min_{(x_{c+1},\dots,x_n)} & \ \omega\, \mathbb{E}I(x_{c+1},\dots,x_n) \\ & + \sum_{i=c+1}^{n} \mathbb{E}W^{(i)}(x_{c+1},\dots,x_n). \qquad (4.19)\end{aligned}$$

From this vector, via the relation $t_{c+i} = \sum_{j=1}^{i} x_{c+j}$, and the fact that all servers start with a customer, i.e., $t_1 = t_2 = \cdots = t_c = 0$, one obtains a regular schedule $\boldsymbol{t}$. Since we study the arrivals of a batch of b customers, we impose some elements of the inter-arrival vector $\boldsymbol{x}$ to be zero:

$$\boldsymbol{x} = \left(\boldsymbol{x}_2^{(b)}, \boldsymbol{x}_3^{(b)}, \dots, \boldsymbol{x}_{\lceil \frac{n-c}{b} \rceil}^{(b)}\right) \quad \text{with} \quad \boldsymbol{x}_i^{(b)} = (x_{(i-2)\cdot b+c+1}, \mathbf{0}) =: (x_i^{(b)}, \mathbf{0}).$$

Note that the vector of zeros in $\boldsymbol{x}_i^{(b)}$ is of size $b-1$, except the last, in which the batch size denoted by $b^{\dagger}$ equals the remaining customers $n - c \mod b$. Hence, that vector $\mathbf{0}$ is of size $b^{\dagger} - 1$. With this restriction in the scheduling framework of Kuiper and Lee (2022) we can use optimization routines to find the optimal schedule for the batches to arrive, which will be given by the vector $\boldsymbol{x}^{(b)} = \left(x_1^{(b)}, \dots, x_{\lceil \frac{n-c}{b} \rceil}^{(b)}\right)$ to create the following experiments to study how the short-run appointment schedules compare to their steady-state, long run counterparts.

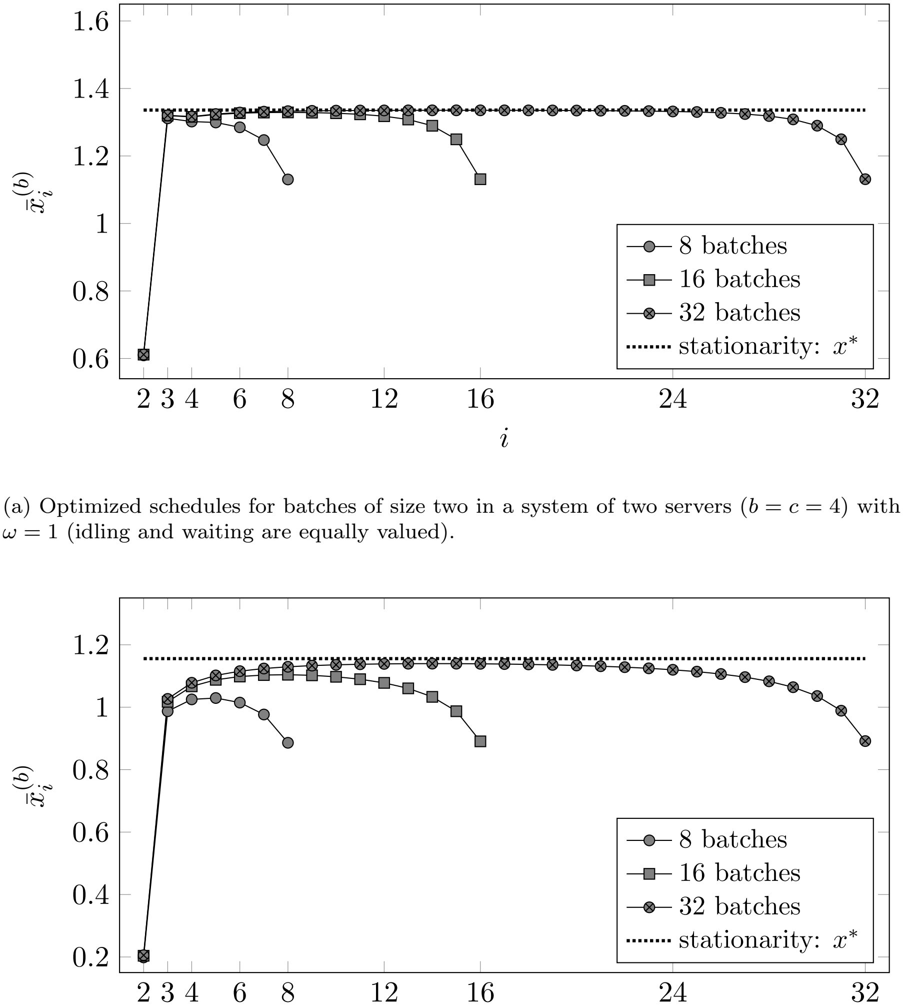
We see from Figures 4.6 through 4.8 that the transient solution quickly approaches the steady state solution, more so for a lower cost of idle time and for larger systems, as might be expected. An interesting artefact that can be seen in Figures 4.7(a) and 4.8(a) is that the second batch to arrive arrives very close to or even after the steady-state solution would suggest and is followed by a very slight dip. This behavior is not replicated when either arrivals or servers are singular. We speculate that when arrivals are singular, more control can



(a) Optimized schedules for batches of size two in a system of two servers ($b = c = 2$) with $\omega = 1$ (idling and waiting are equally valued).

(b) Optimized schedules for batches of size two in a system of two servers ($b = c = 2$) with $\omega = 5$ (idling is valued five times more important than waiting).

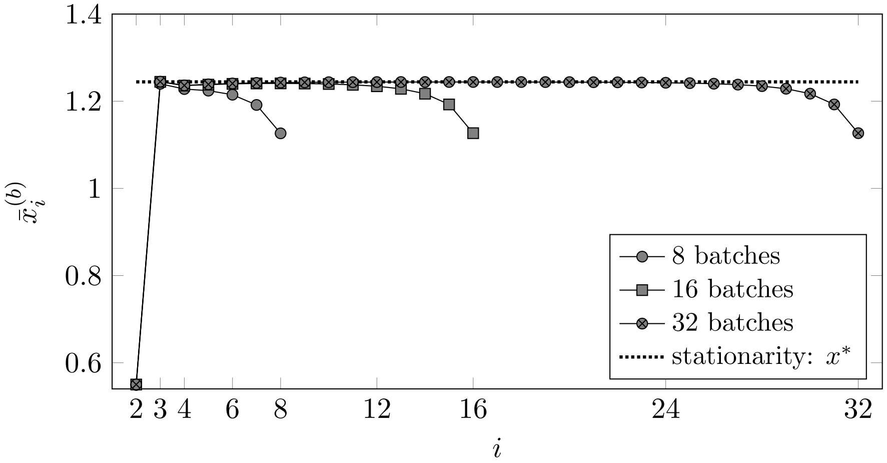
Figure 4.6: Comparison of transient schedules with their steady-state counterparts either from Table 4.2 or Table 4.4. In the top panel the cost parameter is set to $\omega = 1$, while in the bottom panel $\omega = 5$.



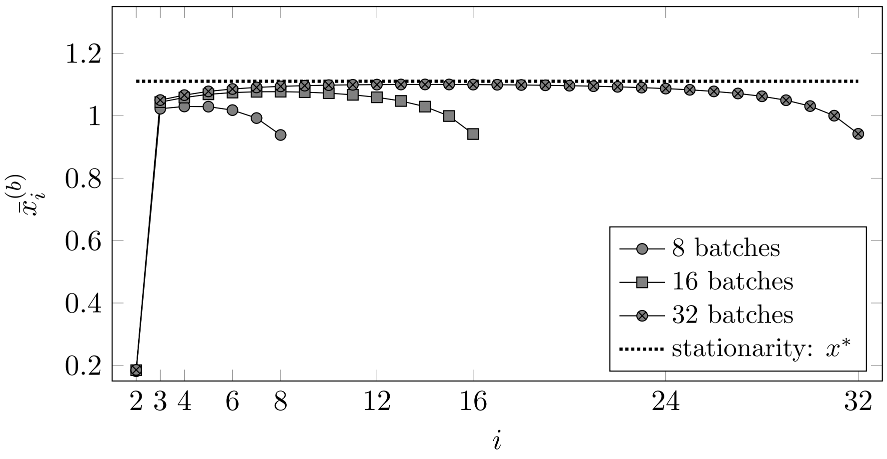
(a) Optimized schedules for batches of size two in a system of two servers ($b = c = 4$) with $\omega = 1$ (idling and waiting are equally valued).

(b) Optimized schedules for batches of size two in a system of two servers ($b = c = 4$) with $\omega = 5$ (idling is valued five times more important than waiting).

Figure 4.7: Comparison of transient schedules with their steady-state counterparts either from Table 4.2 or Table 4.4. In the top panel the cost parameter is set to $\omega = 1$, while in the bottom panel $\omega = 5$.



(a) Optimized schedules for batches of size two in a system of two servers ($b = c = 8$) with $\omega = 1$ (idling and waiting are equally valued).



(b) Optimized schedules for batches of size two in a system of two servers ($b = c = 8$) with $\omega = 5$ (idling is valued five times more important than waiting).

Figure 4.8: Comparison of transient schedules with their steady-state counterparts either from Table 4.2 or Table 4.4. In the top panel the cost parameter is set to $\omega = 1$, while in the bottom panel $\omega = 5$.

be exacted over idle and waiting time by determining the exact arrival epoch of the next customer, and that when servers are singular these can likewise be controlled by deciding when the single server will next become busy. When both arrivals and servers are plural, however, some control is lost as the next batch to arrive must be of size b, regardless of how many servers may or may not be occupied. Thus the first arrival aims to minimize idle time in a sparsely occupied system where waiting time is not a concern, then the next batch to arrive is chosen as to reduce the now increasing waiting time, which is why this phenomenon is not seen when $\omega = 5$. After this peculiarity the schedule returns to its course.

4.4.5 The Case for Large Batches and Time Windows

Table 4.1 indicates that there may be some promise to be found in *large batch* systems, where the number of servers is significantly smaller than the batch size and when utilization is high. For example, for 8 servers and a batch size $b = 32$ the average waiting time reported in Table 4.1 is still less than that of a customer in the singular system. We look at this setting in more detail in Table 4.11, also paying attention to the expected waiting time of the highest numbered batch member.

One of the implicit assumptions of our model is that all customers arrive at the beginning of their time slot. Even with this assumption in place (and so the final customer in the batch must wait for all other customers to be served) it can be seen from Table 4.11 that for a high utilization, the maximum expected waiting time of a batch member ($\mathbb{E}W_q^b$) for $c = 8$ and $b = 32$ or $c = 16$ and $b = 64$ is less than that of even a singular queue.

This also begs the question of what were to happen were we to relax this assumption and open systems up for *time window* optimization. For example, if average service time lasts 10 minutes, then in a system with 16 servers and batches of 64 to arrive, a window of almost 45 minutes would be offered in which these 64 customers could arrive at their own convenience.

		$u = 0.9$		
c	b	x	$\overline{W_q}$	$\mathbb{E}W_q^b$
1	1	1.1111	4.1787	4.1787
8	32	4.4444	1.6616	3.5584
8	64	8.8888	3.4668	7.3741
16	32	2.2222	0.6628	1.5729
16	64	4.4444	1.4942	3.4156

Table 4.11: Mean average and maximum expected waiting times for a singular system versus differing *large batch* systems.

4.5 Discussion and Conclusion

We have concentrated on the design of appointment books for multiple servers that operate in parallel. As studied in several works, such as Zacharias and Pinedo (2017), Soltani et al. (2019), Kuiper and Lee (2022), such systems are prevalent in various scheduling settings, in which a service may be received from any of the employees, e.g., Apple's Genius bar, (COVID) test and vaccination centers, but also other healthcare settings in which continuity of care is of lesser importance. This work amends the traditional set-up of appointment schedules for multiple servers by differentiating between two scheduling paradigms. More specifically, it coined the terms pure and batch pooling to differentiate between two scheduling paradigms that are both capable of reaping the benefits of pooling.

In both batch and pure pooling, the waiting queue is pooled, but the difference lies in the arrival pattern. In pure pooling the arrivals are singular, whereas in batch pooling one schedules as many customers as there are servers. The motivation behind batch pooling is that one hopes to capture many of the established benefits of pooling, while having the strategic simplicity of parallel single-server systems, i.e., a unique appointment book for each server—at least on paper—this helps to have increased ownership and counter adverse effects of pooling as described in Song et al. (2015).

To study the merits of batch pooling, a classical queueing model for exponential servers is enriched to accommodate batch arrivals. In Kendall's notation, we study the $D^b/M/c$ queue in steady state, in which D^b indicates that we have batch arrivals of size b arriving at deterministic, yet-to-be-optimized inter-arrival times x. Aside of the special case of batch pooling ($b = c$), the framework permits the computation of expected waiting times under any configuration of batch size, number of servers, and inter-arrival times. Proving that for parallel servers and any batch size the expected waiting times are convex in x—an open problem in literature—we leverage our framework in an objective function composed of expected waiting and idle times to establish optimal schedules.

We find that batch pooling has many of the server-side benefits of pure pooling. While not as beneficial to the customer on average as pure pooling, its expected waiting times are significantly better for the vast majority of customers than in the single-server case; especially at high utilizations. Also, the analytical model is expanded to include no-shows and walk-ins, which indeed confirm the heuristic that if no-shows occur with a probability of q, inter-arrivals should be shrunk with a factor $(1 - q)$ based on the assumption that a no-show can be modelled as a customer with zero waiting time, a heuristic that is frequently used, e.g., in Çayırlı et al. (2012).

Adapting the methodology found in Chapter 3 we are able to study batch arrivals in transient scenarios. This allows us to verify our results, showing that solutions quickly converge to steady state, while featuring the well-known dome-shape pattern (Hassin and Mendel 2008, Kuiper et al. 2015). Studying these schedules in more detail uncovers an unusually higher second inter-arrival, which arises because of the inflexibility that the workload of an entire batch brings to an empty waiting queue of a multi-server system. Furthermore, as transient solutions tighten toward the end, these imply longer waiting for customers scheduled at the session end. This might have negative consequences, like customers balking or abandoning the queue, and thus essentially becoming no-shows, a setting studied in Zhang et al. (2022). Our solutions can also serve as a remedy, because they balance idling against waiting in steady state.

As at the arrival of a batch the expected waiting of each batch member can be computed, we compared the expected waiting times of each batch member to that of *pure pooling*. We saw that the lowest numbered batch members had a waiting time less than that of the pure pooling customer, while the highest numbered batch members had a waiting time in excess of the pure pooling customer's. This has managerial implications for systems with triage, where customers are upon arrival assigned a position in the batch according to the severity of their situation. We also compared individual expected waiting times with those in a single-server system, finding an improvement for almost all customers.

Since imposing batch arrivals reduces flexibility, increasing the batch size always comes with additional costs. Even for batch pooling, in which the batch size grows with the number of servers, at around 20 servers such an appointment book becomes inferior to organizing it as a walk-in clinic where walk-ins are modelled as Poisson arrivals. This suggests that for large systems it may be attractive simply not to optimize.

Besides extending the model to deal with other service-time distributions, a practical consideration is that of unpunctuality. One can imagine customers not to arrive all strictly on time. Such an extension would be to consider the relaxation to more 'open' systems, in which there is a time window in which customers are scheduled to arrive. Still, if one restricts unpunctual customers to arriving at another batch's scheduled moment, one can up to some degree model a customer too late or too early as a no-show for his or her arrival moment, while a walk-in at another. In all, this work introduces a comprehensive analytical framework to study and optimize different appointment book designs in terms of number of servers and batch size, and allows the integration of several relevant features in the field of appointment scheduling.

Chapter 5

On Scheduling Operating Rooms

5.1 Introduction

This case report is carried out at the Red Cross Hospital in Beverwijk (RKZ), the Netherlands. It concentrates on the design and implementation of a new master surgery schedule (MSS). This is a 4-week cyclic schedule that allocates time to surgical specialties. As part of the surgical suite, operating rooms are scarce, expensive, and vital resources; see May et al. (2011), Childers and Maggard-Gibbons (2018) and Jung et al. (2019). Their use is of primary interest to hospitals and has spawned a rich literature presenting various models and optimization methods, see the influential review by Cardoen et al. (2010). Besides some notable exceptions, the literature is largely concerned with the development of theoretical models and advancements; introducing additional features, proposing more efficient approaches, or overcoming new theoretical challenges. Although these features and challenges are often inspired by practice, only a handful of works show adoption of operations research in practice, see also Samudra et al. (2016). This work reports on a case in which operations research was used to create a new MSS that, after adoption by the hospital, improved productivity considerably. Besides focusing on the finer points of the model, we outline the path followed to successfully apply operations research in the operationally and politically complex healthcare context.

5.1.1 Motivation: Development of a new MSS

In 2006, the Dutch healthcare market was radically reformed by the implementation of a new law to create regulated competition. The goal was that citizens would choose insurers and healthcare providers based on the quality of care, service and price. Under the new law the growth of total healthcare costs was restricted, and the administrative burden increased significantly, see Jeurissen et al. (2021). During the COVID-19 crisis, this limitation in growth was felt all the more harshly as the burden on healthcare increased and many procedures had to be postponed. However, while affordability is needed to restrict the substantial growth of expenses in the future, hospitals compete with one another for qualified personnel in a struggle to meet annual production agreements with insurance companies.

In 2022, the Dutch government acknowledged that the healthcare market in the Netherlands was under pressure and published an integral healthcare agreement stating several challenges to overcome. A shortage of staff limits accessibility to and quality of healthcare. The requirement for affordability restricts the growth of future expenses, while at the same time hospitals must compete with one another for qualified personnel in a struggle to meet annual production targets agreed upon with insurance companies. As a result, one of the main areas for improvement falls on efficient use of resources, as reported by the Dutch Ministry of Health and Sport (2022).

The effects of the pressure on the healthcare market were noticeable in the Red Cross Hospital (*Rode Kruis Ziekenhuis*, RKZ) in Beverwijk in the Netherlands, where the surgical suite makes up just more than 30% of the revenue. There is a direct relationship between waiting times and the utilization of key resources such as staff and operating rooms. Shortages of staff leads to increased waiting times, and hiring more personnel or opening more facilities leads to financial concerns as the hospital needs to meet the production targets negotiated with insurers, which are based on the size of the hospital. As a consequence, hospitals are focused on matching the production targets in the most efficient way, as staff shortages persist.

The emphasis placed on efficiency prompted RKZ to start development of a new MSS that would make efficient use of operating room time while ensuring quality of care. Concurrently, sufficient time should be reserved to deal with acute patients, i.e., emergency patients. Lastly, internally, it is preferred that the outflow of patients is fine-tuned such that the workload of supporting personnel, such as nurses, is stable.

5.1.2 Contribution: Practice-oriented Research

RKZ maintains a 4-week cyclical block scheduling system, in which two specialties may share an operating room on the same day. This induces *turnaround costs*, which RKZ wants to minimize, while also fine-tuning the outflow of patients as mentioned above. As commonly done, we model the problem as a Mixed Integer Linear Program (MILP), presented in Section 5.7. We additionally break the problem into two stages to mitigate the problem of symmetry inherent in the formulation of the MSS optimization problem, while incorporating a multitude of practical constraints.

In finding a fitting solution for the scheduling problem, we discovered that this was one of the few successfully applied projects in this field and could contribute to an underdeveloped part of operations research. Therefore, we organized the report such that it can be read as a roadmap for successful healthcare optimization using operations research. The value thus lies in bringing the operations research and medical professional communities closer together, bridging the gap of theory and real-life implementation, articulated in Cardoen et al. (2010) as: *"[...] we encourage the provision of additional information on the behavioral factors that coincide with the actual implementation. Identifying the causes of failure or the reasons that lead to success may be of great value to the research community."* Medical professionals struggle when making schedules manually, and the quality of the solution is low in the case of complex schedules. Operations research practitioners grapple to understand the needs of hospitals: the variety of constraints, the objectives and the way to implement a new schedule.

Solving a practical problem requires dealing with multiple stakeholders, ambiguity about requirements and objectives, organizational struggles, and other barriers. We found the inclusion of a domain expert, the hospital's capacity manager, invaluable in enabling the project. The capacity manager was able to critically assess requested constraints and to provide an extensive internal evaluation of the project in Section 5.5.3.

We reaffirm that the operations research model is seen as an objective decision-maker, which simplifies the adoption of a solution in a politically driven environment. To aid in decision-making we provide at each design round a Pareto plot, which presents various scenarios and makes it easier for management to decide between schedules. We also found that there should be sufficient time reserved for these design rounds; here, nearly half a year. Lastly, an impetus helps to ensure adoption, for example, a drastic change in case mix since the adoption of the last MSS, mainly as this guarantees commitment of hospital management, see for example Zenteno et al. (2016).

The paper is structured as follows. First, we outline relevant literature on the optimization of surgery scheduling, with an additional focus on works that apply operations research in healthcare. Then, in Section 5.3, we provide the context of the case and show how the environment is translated to objectives that can be captured as an optimization problem. Next, we show how scheduling constraints and requirements are obtained and grouped, which are all put in a mixed integer linear program, which is outlined in Section 5.4, the mathematical details of which are postponed to Section 5.7. In Section 5.5, we focus on the implementation process that culminated in the new MSS; over two design rounds, the model was further tailored to practice. Finally, in Section 5.6, we reflect, conclude and provide recommendations for the application of operations research in healthcare.

5.2 Literature Review

There is a vast amount of literature on surgery scheduling. Therefore, we decided to scope our review to three themes that help position our contribution. We first consider general planning and control decisions on different levels in which we embed our case about the master surgical schedule (MSS). Second, we outline some classical approaches developed to solve this type of problem and, third, we provide an overview of the few articles that applied operations research in practice.

5.2.1 Planning and Control of the Surgical Suite

Considering the planning and control of the surgical suite, there have been various influential overviews, for example: Cardoen et al. (2010), May et al. (2011), and Hulshof et al. (2012) all provide comprehensive overviews of the decisions revolving around surgical care. Hulshof et al. (2012, Section 3.3) provide a detailed overview of the strategic, tactical, and operational dimensions of surgical care. Relevant to this project, they place *case mix* at the strategic level; see also the first problem as defined by Gupta (2007). Within the tactical level Hulshof et al. (2012) place *patient group identification*, *time subdivision* (e.g., *MSS*), and *staff shift scheduling*, placing this project at the tactical level.

Before we delve deeper into the details of the MSS, we next consider the operational level, wherein we find two streams of literature *surgical case scheduling* and *emergency case scheduling*; the allocations of specific surgeries to a specific time and place. This is called the surgery scheduling problem in operations research, see Zhu et al. (2019), and aligns with Problems 2 & 3 as described by Gupta (2007), comprising the scheduling and sequencing of patients. These decisions are not within the scope of this project, but revolve around minimizing under- and overtime; see, among others, Denton et al. (2007), May et al. (2011). The focus on under- and overtime in surgery scheduling also has a strong connection with the allied problem of appointment scheduling, which focuses on minimizing idling and waiting, as discussed in Chapter 2 of this dissertation, also published as Lee and Kuiper (2024).

Besides considering planning and control elements, one can also improve the surgical suite by the use of Lean as discussed by Kim et al. (2006) or the combination of Lean Six Sigma as in De Koning et al. (2006). These approaches are often used as improvement methodologies to improve efficiency by, among others, eliminating waste in processes. Although there are some critics about the use of these methodologies in healthcare, see Radnor et al. (2012) and in relation to COVID-19 outbreak (Kuiper et al. 2022), there have been considerable successes in improving the performance of the surgical suite by using Lean. Both Harders et al. (2006) and Collar et al. (2012) report significant reductions of approximately 20 minutes of non-operative time between two operations. In addition, it minimizes the risk of going beyond the targeted session-end time, e.g., 5 PM, potentially reducing overtime pay, but already by reducing changearound times, it is estimated that labor costs are reduced significantly, see Dexter and Epstein (2005). It is also known via Dexter et al. (2019) that switching from one specialty to another increases the mean turnaround times, and thus, it is preferred to have one specialty assigned a block.

So, with regard to the organizational decision hierarchy, this project about the design of a new MSS belongs to the tactical level. It starts right after the case-mix allocation—a strategic decision—which is a key input for the new schedule. An MSS allocates blocks (capacity) to clusters, a sub-group of specialties, but does not a priori prescribe which surgeries have to be executed; this is left to each assigned cluster to decide, still allowing each cluster to have its own autonomy and flexibility to use the allocated time, as noted by Van Oostrum et al. (2010). This aligns well with the classical structure in which hospitals have independent physician-entrepreneurs, who get paid a fixed fee for each procedure—a situation that is typical in healthcare systems in Western countries, see also Blake and Donald (2002) who describe it in the wording "the agency relationship is sacrosanct." This type of problem is not separately identified as one of the key problems in Gupta (2007), as it is only referred to as providing OR time allocation to specialties; thereby, it is integrated with determining the case mix problem.

5.2.2 MSS Optimization in Practice

As articulated by Cardoen et al. (2010), much of the work seems focused on practice, but remains unclear whether it is and how it is put into practice; they explicitly state: *"Even if the implementation of research can be assumed, authors hardly provide details on the process of implementation."* A good reason for this is that there is an unclear distinction between theory-oriented and practice-oriented articles, see for example Table 13 of Samudra et al. (2016). Also, in a recent review by Wang et al. (2021), it is highlighted that future work should focus on narrowing the gap between theory and practice. The value of using an MSS as an advance planning tool is highlighted in Van Oostrum et al. (2010). Also, the authors outline possible obstacles that hinder the implementation of a (new) MSS in a hospital, such as the degree of specialization, resistance, and leadership. Scanning the literature, there are many studies in healthcare that are inspired by practice but are not applied, for example, Denton et al. (2010a), which is motivated by real problems at Mayo Clinic in Rochester, Minnesota. In their works, Beliën et al. (2009) van Essen et al. (2014), and in a similar vein Guido and Conforti (2017), use a testbed to demonstrate the potential improvement of using their solutions. Finally, we mention the work of Marques et al. (2019), who also ask for the input of the decision maker, after which they optimized the MSS of a medium-sized Portuguese hospital: *The head doctor of the surgical suite was particularly impressed with the study undertaken. He appreciated the solutions at first glance but a deep understanding requires further analysis which is facilitated by this kind of methodology. The implementation of this approach in the hospital is currently under discussion.* It underpins that applying operations research in practice is more than presenting your methods, solutions, and improvement potential. Such an approach does not pave the ground for actual implementation. Typically, as noted in Delesie (1998), the operations research community favors putting forward their models, as we also see in the surgery scheduling literature. However, practice does not simply adjust to a stylized model, and practitioners do not conceptualize according to these ways, so instead, operations researchers can better "lend their ear" to developing and tailoring a model according to the hospital's needs.

We now proceed to give an overview of actual *applied* works in the field of optimizing MSSs. From these various papers, we have extracted common themes that are subject to optimization and have summarised these themes in Table 5.1. *Ward workload* is one such theme, present in Zenteno et al. (2016) and Benchoff et al. (2017) who both aim to minimize congestion in wards, particularly by means of minimizing peak demand, while Visintin et al. (2017) and Vanberkel et al. (2011) aim to meet a target utilization, Visintin et al. (2017) by means of goal programming and Vanberkel et al. (2011) via a series of informed swapping decisions and inspection of the resulting ward occupancies. They also report that they applied variants of their model in three other hospitals. *Operating room utilization* is another common objective for which Visintin et al. (2017) aim to directly maximize utilization, while Zenteno et al. (2016), Benchoff et al. (2017) and Vanberkel et al. (2011) report improvements in the ward loading, leading to a decrease in the number of surgery cancellations. The third common theme is the distribution of *case mix hours*, which is an objective for Blake and Donald (2002), who aim to give each specialty the minimum number of operating hours they are promised, while this theme is considered a constraint by the other papers. Zenteno et al. (2016), Benchoff et al. (2017) and Visintin et al. (2017) also include other objectives in their models, but these are specific to each paper and do not constitute common themes. We have also extracted themes which are not subject to optimization but are nonetheless interesting. We consider the *scale* of the problem, that is, how many operating rooms and specialties were considered in each case; *schedule type*, i.e., whether the schedules devised were cyclic or covered a planning horizon. As Blake and Donald (2002) do not optimize for ward workload, they do not consider the modeling of bed occupancy at all.

Another theme in the literature in both theoretical and applied works is bed occupancy, which describes how many patients occupy beds in a ward at a given point in time, which is crucial in determining ward workload. We break bed occupancy up into two required pieces of information to be modeled: *outflow*, the number of patients operated on in each block who will then be discharged to a ward; and *length of stay* or *LOS*, how long a given patient lies in a ward. A simple approach would be to take both outflow and LOS in expectation, yielding an expected bed occupancy. When more accuracy is desired, however, a probability distribution can be calculated or assumed for either or both outflow and LOS, yielding either finer-grained conditional expectations or even full probability distributions for bed occupancy.

Vanberkel et al. (2011) give the most detailed modeling of bed occupancy, using average outflows from procedures and applying to these a binomially distributed length of stay to achieve *a distribution* over the number of beds needed on any given day of the schedule. Benchoff et al. (2017) use average outflows from procedures, employing empirical distributions for the lengths of stay of different patient types to arrive at the *expected* number of occupied beds on any given day of the schedule. Zenteno et al. (2016) stratify patients into groups based upon their length of stay (seemingly in expectation) and calculate the expected number of days a bed will be occupied for each surgical block assigned. Unfortunately, we were not able to discern whether outflow and LOS were modeled in expectation or via probability distributions. Visintin et al. (2017), create a scheduler for a short planning horizon and so make an entry in the schedule for each patient. They therefore know beforehand what the outflow will be. They model the length of stay with a discretized probability distribution over the number of days a patient lies in a ward. These values were then used in a simulation to test the robustness of the solution to variation in bed occupancy.

5.3 Case Background

We will now present details of the forces which lead to the undertaking of this project. This serves as a preliminary step in which the context under which the project is undertaken is analyzed, see also Coughlan and Coghlan (2002). We will then translate these into objectives important to the hospital. Finally, in this section, we discuss the formulation of the case mix and the patient group identification, which were inputs into the operations research model.

RKZ has seven operating rooms, of which one is permanently reserved for burned skin treatment, an urgent type of care for which surgery-duration predictions and plannings are hard to make. For this reason, this room is left out of scope for this project. In fact, the hospital has a national reputation for being one of the few Dutch hospitals that treats burned skin, a medical field in which general and plastic surgeons work together. Aside from this function, the hospital also provides regional care. Available specialties for surgery are general surgery, orthopedics, plastic surgery, urology, gynecology, otolaryngology, orthognathic-surgery, and neurosurgery.

Figure 5.1 depicts the general flow of patients receiving surgery in RKZ. Patients are first administered anesthetics before entering one of the six surgery rooms. Each day, these rooms run *morning* (08:00 - 12:30) and *afternoon* (12:30 - 17:00) surgery blocks, which combined we call a whole block. Each day one room's afternoon block is shortened to keep time available for emergency patients. Afterwards, surgery patients move on to the Post Anaesthetic Care Unit (PACU), and later to one of the available wards (*outflow*). Most surgical patients go to the daycare or clinical wards where patients from several specialties are treated together.

5.3.1 Concerns of the Hospital

The hospital wished to meet production targets set by insurers while balancing the supply of and demand for post-operative care in order to improve the quality of patient care and stabilise the workload of personnel. This latter goal would be achieved by *balancing outflow* of patients from surgery to the wards. To this end, a new MSS was desired which would replace the previous MSS in

Work	Ward workload	OR utilization	Case mix hours	Scale	Outflow modeling	LOS modeling	Bed occupancy	Schedule type
Blake and Donald (2002)	Not considered	Not considered	*Objective*	10 ORs, 5 specialties	Not considered	Not considered	Not considered	Cyclic (1 week)
Vanberkel et al. (2011)	*Objective*	Implicit	Constraint	6 ORs, 6 specialties	In expectation	In distribution	In distribution	Cyclic (1 week)
Zenteno et al. (2016)	*Objective*	Implicit	Constraint	56 ORs 150 surgeons[a]	Unknown[†]	Unknown[†]	In expectation	Cyclic (4 weekly)
Benchoff et al. (2017)	*Objective*	Implicit	Constraint	15 ORs, 15 specialties	In expectation	In distribution	In expectation	Cyclic (4 weekly)
Visintin et al. (2017)	*Objective*	*Objective*	Constraint	7 ORs 15 specialties[b]	Known ahead of time	In distribution	In distribution	Planning horizon (2 weekly)
This work (2023)	*Objective*	*Objective*	Constraint	6 ORs, 29 specialties[c]	In expectation	In expectation	In expectation	Cyclic (4 weekly)

Table 5.1: Common themes from applied literature: *Ward workload* and *OR utilization*, and *Case mix hours* of surgical clusters. *Objective* and *Constraint* indicate the role of this theme in the problem; *Implicit* is used either when this theme is aided by another or where it is not mentioned, but cannot be assumed to have not been included via some constraint. [a] Zenteno et al. (2016) do not report on the number of surgical specialties. [b] These are further divided into almost 150 clusters based on patient group surgery time and length of stay characteristics. [c] These are in fact *clusters* as described in Section 5.3.3. [†] We were not able to discern with certainty whether an expectation or distribution was employed.

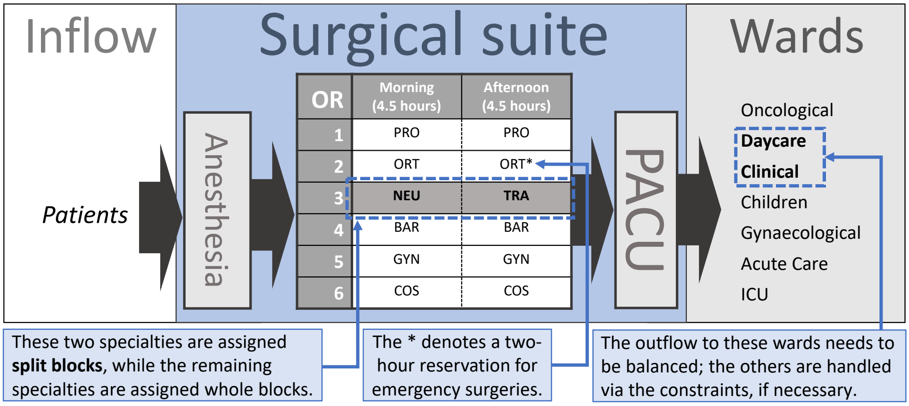

Inflow
Surgical suite
Wards
Patients
Anesthesia
OR
Morning (4.5 hours)
Afternoon (4.5 hours)
1 PRO PRO
2 ORT ORT*
3 NEU TRA
4 BAR BAR
5 GYN GYN
6 COS COS
PACU
Oncological
Daycare
Clinical
Children
Gynaecological
Acute Care
ICU
These two specialties are assigned **split blocks**, while the remaining specialties are assigned whole blocks.
The * denotes a two-hour reservation for emergency surgeries.
The outflow to these wards needs to be balanced; the others are handled via the constraints, if necessary.

Figure 5.1: An overview of the setting and problem description for a single day; the full schedule consists of 20 days.

use since the end of 2018 (see Table 5.2). The new MSS was to be implemented by the end of 2022, with its two primary objectives being:

1. *Production:* There were 40 split blocks (as defined in Figure 5.1) in the old schedule which caused 20 turnarounds; this can be seen in the gray shaded cells in Table 5.2. Split blocks necessitate the refitting of operating rooms from being suited for one cluster to operate into being suited for another. These incur direct costs from refitting and also carry the opportunity cost of production as they eat into time that could otherwise be spent on procedures. Furthermore, as schedulers typically schedule surgical cases conservatively to avoid overrunning a block, more split blocks results in more unutilized time. Allocating whole blocks in the new MSS will minimize total turnaround time and therefore increase productivity.

2. *Balancing Outflow:* The outflow of the old schedule to subsequent wards was not well balanced, meaning that the number of patients admitted per day deviated too much from the desired pattern. Deviations, as can be seen in Figure 5.2, can lead to a reduction of quality and speed of care for patients and create stress and dissatisfaction for staff, or lead to inefficiencies if resources go unused. To match supply and demand for nursing capacity, two approaches are possible: adjusting the patient inflow or modifying the nursing schedule. Due to physical limitations, this optimization employed the first approach. Three wards share this problem.

   a. The occupation of the clinical ward should be stable over the weekdays and weeks. A high and steady inflow on Mondays helps to stabilize occupation. Length of stay is presumed constant for all patients per ward, as data analyses have shown that there are only subtle differences in LOS for most specializations.

b. The daycare ward combines surgery and non-surgery inflow. This ward experienced problems with too many or few patients on specific days. The sum of non-surgery patients varies strongly, but is steady per day and week. For this ward a target number of patients was desired, assuming a fixed number of non-surgery patients per day.

c. In 2021, a new ward was opened with highly specialized staff for oncological care. This ward is limited in size and facilitates a specific group of patients after surgery. In the old schedule, some days generated problematic outflow: surgeries involving cancer (such as: breast, bladder, and colon surgery) that take place on the same day, were required to be better spread out in the new schedule.

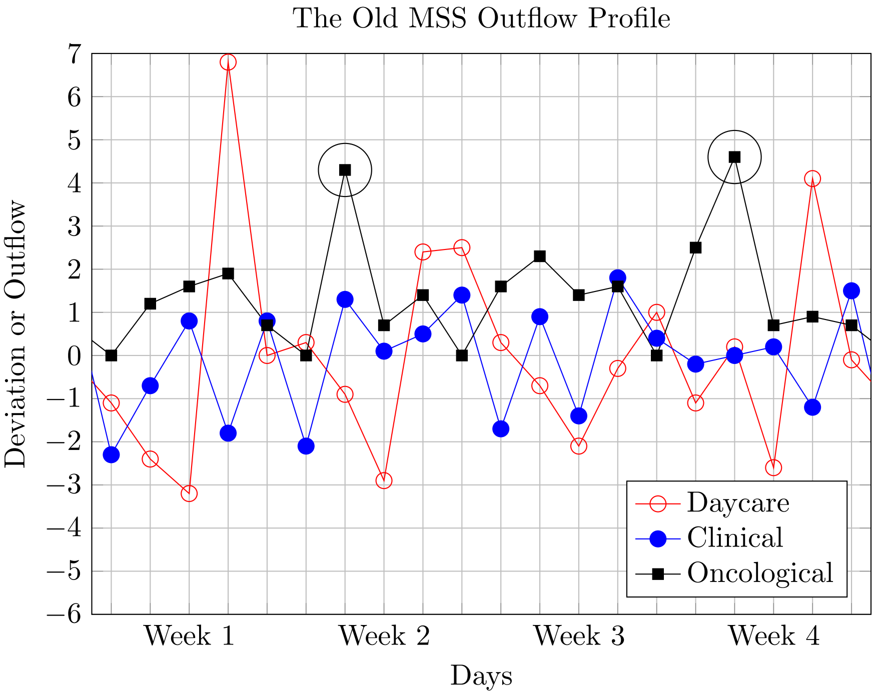


Figure 5.2: The corresponding outflow deviations from the target of the MSS for daycare and clinical as well as the net outflow to the oncological ward, which should ideally be well below 4. Two violations of this principle are circled.

		OR1	OR2	OR3	OR4	OR5	OR6
Week 1	Mon	PRO PRO	ORT ORT*	NEU TRA	BAR BAR	GYN GYN	COS COS
	Tue	BRE BRE	ORT ORT	ENT ENT	MAS GEN	SWI SWI*	PLA_Z PLA_Z
	Wed	URO URO*	ORT ORT		BAR BAR		PLA_T PLA_T
	Thu	ENT URO	ORT ORT	ORG ORG	GEN GEN	IHE IHE*	PLA_W PLA_W
	Fri	TRA TRA*	ORT ORT	MAS VAS	BAR BAR	GYN GYN	PLA_L PLA_L
Week 2	Mon	SWI SWI	ORT ORT	ENT ENT*	BAR BAR	NEU TRA	PLA_M PLA_M
	Tue	URO URO	ORT ORT	MAS MAS	GEN GEN	TRA TRA*	PLA_Z PLA_Z
	Wed		ORT ORT*		BAR BAR	MAS VAS	PLA_T PLA_T
	Thu	NEU GEN	ORT ORT	ORG ORG	GEN GEN	ENT ENT*	PLA_W PLA_W
	Fri	COS COS	ORT VAS	CHC TRA	BAR BAR	GYN GYN*	PLA_L PLA_L
Week 3	Mon	URO URO*	ORT ORT	CHC TRA	BAR BAR	PLA_T PLA_T	COS COS
	Tue	BAR TRA	ORT ORT	ENT ENT*	GEN GEN	MAS MAS	PLA_Z PLA_Z
	Wed		ORT ORT*	MAS MAS	BAR BAR	GYN GYN	PLA_T PLA_T
	Thu	URO URO*	ORT ORT	ENT NEU	GEN BAR		PLA_W PLA_W
	Fri	SWI SWI*	ORT VAS	IHE TRA	BAR BAR	GYN GYN	PLA_L PLA_L
Week 4	Mon	URO URO	ORT ORT	GEN NEU	BAR BAR	TRA TRA*	PLA_M PLA_M
	Tue	URO URO	ORT ORT	ENT ENT*	GEN MAS	MAS MAS	PLA_Z PLA_Z
	Wed		ORT ORT	VAS MAS	BAR BAR		PLA_T PLA_T^*
	Thu	SWI SWI	ORT ORT	ORG ORG*	GEN GEN	ENT ENT	PLA_W PLA_W
	Fri	COS COS	ORT VAS	MAS TRA	BAR BAR	GYN GYN	PLA_L PLA_L^*

Table 5.2: The old MSS, in which the asterisks ($*$) indicate which OR is dedicated to emergency surgeries.

5.3.2 Setting the Case Mix

Insurance companies dictate yearly production for hospitals by setting a limit on the maximum financial compensation a hospital receives. To meet these targets, RKZ runs 28 surgery blocks per week. The case mix divides available surgery time over specialties. Formulating a division of case mix time was a combination of historic data analyses, expectations for the future, and company strategy. Data analyses showed occupation and utilization of surgery rooms and trends in waiting lists helped to reveal future demand for specialties. As a result, orthopedic and plastic surgery hours were adjusted to fit future demand. The output of the case mix is the number of surgery hours assigned to each specialty per cyclical schedule of 4 weeks. Besides determining the case mix by a detailed analysis, one can also solve for economic considerations by using an operations research model, as in Blake and Carter (2002).

5.3.3 Identification of Patient Groups

Within specialties, there is a large variety in resource utilization, including the medical knowledge of the surgeon, the need for medical equipment, the ward to which the patient is discharged after surgery and the number of patients that are discharged to that ward. The purpose of patient group identification is to create clusters that are similar in the use of capacity, for more details, we refer to Schneider et al. (2020). Here, out of *eight* surgical specialties, 23 different clusters were initially distilled, later expanded to 29. These clusters were primarily based on the medical knowledge of the surgeons and the wards to which patients would outflow. As an example, *bariatrics* and *proctology* share the same surgeons but are considered different clusters within specialty general surgery because outflow varies: *bariatrics* typically generates *five* clinical patients and *proctology* about *ten* daycare patients.

Before advancing to the optimization of the MSS, the case mix hours per specialty needed to be further divided over the newly-created clusters, this process is called time subdivision by Hulshof et al. (2012). Whereas case mix division was largely chosen by hospital management, surgeons had more say in this time subdivision. To aid in this process, suggested time subdivisions were made based on calculated surgery room occupations and throughput times.

5.4 Project Definition

Here, we describe the project definition and how all input needed for the mathematical model was obtained. Higher-level objectives, as given at the end of Section 5.1.1, for example, efficient production and stable outflow, needed to be translated into quantitative performance dimensions at the operational level. Second, the constraints of the stakeholders, the validity of which were assessed by weighing their influence, were grouped in order to keep the model structured and organized.

5.4.1 Choosing an Objective Function

In the case background in Section 5.3 we saw the two objectives of firstly decreasing the number of turnarounds in order to increase production and secondly balancing the outflow to the three wards (clinical, daycare, and oncological) to match nurse capacity. These two objectives are at odds: by decreasing the number of split blocks we have less opportunity to refine outflow. Therefore, we minimize a weighted combination of two objectives: production can be maximized by minimizing the number of split blocks, while a balanced outflow can be achieved via goal programming.

It was determined that a patient assigned to the clinical care ward cost 3.3 times as much as a patient assigned to daycare. One large deviation from the target outflow was considered proportionally worse than many smaller deviations (even when total deviations are equal), therefore larger deviations were penalized more harshly than smaller deviations. Due to the limited number of options, the goal for the stability of outflow in the oncological ward was turned into a constraint. How exactly the outflow imbalance was modelled for incorporation into the mixed integer linear program is handled in more detail in Section 5.7. Multiple optimization runs were performed using various weights placed on the minimization of split blocks so that different trade-offs between production and outflow balance could be compared.

Outflow to wards was balanced by minimizing the deviation between the number of patients admitted to a ward per day and a desired target. Each time a cluster operated, the expected number of patients who would be discharged to either of these wards was taken. Variation in length of stay was not considered as during a previous simulation study at RKZ length of stay was found to be consistent. Therefore, it was decided to model lengths of stay deterministically, using average values for the lengths of stay of patients in the clinical and daycare wards. The end result is an objective function where each day has its own target outflow per ward independent of other days.

5.4.2 Gathering Constraints from Stakeholders

Constraints were generated by collecting the requirements and wishes of stakeholders. Surgeons, surgery support staff, nurses, and managers were all consulted on what they felt should be included in the model. This process presented a number of challenges. For the vast majority of stakeholders, this was their first encounter with mathematical optimization. This unfamiliarity resulted in stakeholders not immediately knowing what requirements to report, with them presuming that many important restrictions would be implicitly met. Another closely related issue to this is that stakeholders would overlook the literalness of mathematical constraints. Both of these points can be illustrated with an example: a cluster which reports it can work on Tuesdays does not imagine that a mathematical program would schedule them to work only on Tuesdays if that minimizes the objective (this occurred in one early draft of the schedule) and would not immediately think to report that they are only happy to work on Tuesdays up to an as-yet undetermined limit. This example prompts a third difficulty: from the perspective of the modeler, the constraints appeared to be constantly shifting.

Furthermore, the operational structure of a hospital does not always lend itself well to mathematical formulation, for example the cluster ENT consisted of 3 surgeons who all need to be scheduled for a minimum number of hours. This can be accommodated by creating additional clusters, which was carried out, bringing the total number of clusters from 23 to 29.

Other challenges lie within the realm of change management. This project began with a revision of case mix, resulting in a reduction of surgery time for some specialties as well as knock-on effects such as the requirement to redesign outpatient department schedules. This contributed to unwillingness to cooperate with the creation of a new schedule, making it difficult to gather information about constraints. For this reason, conversations on case mix and the new MSS were kept distinct where possible to avoid opinions about the one hindering discussion of the other. Additionally, allowing everyone to include any demand would have limited the possible outcomes of the model and thus the savings attainable. A strict attitude towards demands was thus needed, and the motivation behind them had to be uncovered. For example, weekday availability based upon working in another hospital was considered a valid argument, but reluctance to changing days-off was not.

To mitigate the above problem, constraints were divided into *must-have*s and *would-like*s. Must-haves became constraints in the model, while would-likes, if satisfied by chance, were considered "the icing on the cake". Eventually many staff also saw the benefit of being able to impose mathematical constraints, for example to smooth workload in a predictable manner or to constrain peak demand for their services. Finally, medical staff were convinced that the new schedule would reduce surgery room turnarounds and better match the inflow of patients, resulting in improved waiting lists.

5.4.3 Grouping Constraints

Consulting all stakeholders lead to a large list of constraints describing the current state of the hospitals' capacities. Hereby an important note should be made that capacities can be expanded to lift constraints. Equipment can be bought, and extra people can be hired if the improvements outweigh the costs. For some restrictive constraints, this option was kept on the table to be explored further (see Section 5.5.1). All constraints were grouped into the following classes:

- *Availability constraints* specify that a cluster can operate a block on a given week, weekday and time of the day.
- *Concurrency constraints* forbid two clusters from operating at the same time. E.g., gynecology and urology cannot operate at the same time

because they use the same scopic equipment.

- *Case mix hours* dictate the minimum of OR hours per cluster per cyclic schedule as agreed in the case mix.
- *Similarity constraints* require clusters scheduled to a given block in one week to be scheduled to that same block a number of weeks, usually two weeks, later. This helps to increase the throughput time of oncology patients by simplifying the number of outcomes for the OR schedule.
- *Other constraints.* Include bounds on capacity such as laparoscopic kits and rolling horizons to facilitate evenly spread-out capacity for semi-urgent clusters in the form of minimum capacity per week.

5.5 Implementation Process

A mathematical program, as the one described in the previous section, simplifies the real world. Thereby, it does not fully account for all elements that play a role in the surgical suite. However, it does not have to; the prerequisite is that a model is sufficiently comprehensive, such that its solutions are useful in practice; see also the seminal work of Box (1976). To come to such a model, we followed a timeline as given in Figure 5.3. After the initiation and definition of the project, we followed several design rounds or cycles, as in Coughlan and Coghlan (2002). These rounds were particularly helpful because, on the one hand, they enriched the model with the right, critical, and practical elements to make it realistic and useful. On the other hand, we found that it introduced and paved the road for the broad acceptance of the new MSS, leading to the approval of the final schedule in May 2022. After which the new MSS was implemented and later evaluated. Operations research professionals from the university partook from phases 'Project Definition' until 'Approval'.

5.5.1 The First Design Round

The information in terms of objectives and constraints, as gathered and formulated in the previous section, were modelled in Python. Using a state-of-the-art solver, i.e., *Gurobi*, solutions were obtained when different weights were chosen for penalizing split blocks. The possible solutions are illustrated in Figure 5.4 and form a so-called Pareto frontier, no better solution can be obtained for this specific trade-off of number of split blocks (x-axis) and outflow deviation (y-axis). In addition, some specific scenarios were run, which were known to be some open questions and are magnified in the plot. The models in the frontier have 2540 variables and around 3269 constraints. Note that schedules below the frontier have fewer constraints, whereas solutions above impose extra restrictive constraints, e.g.:

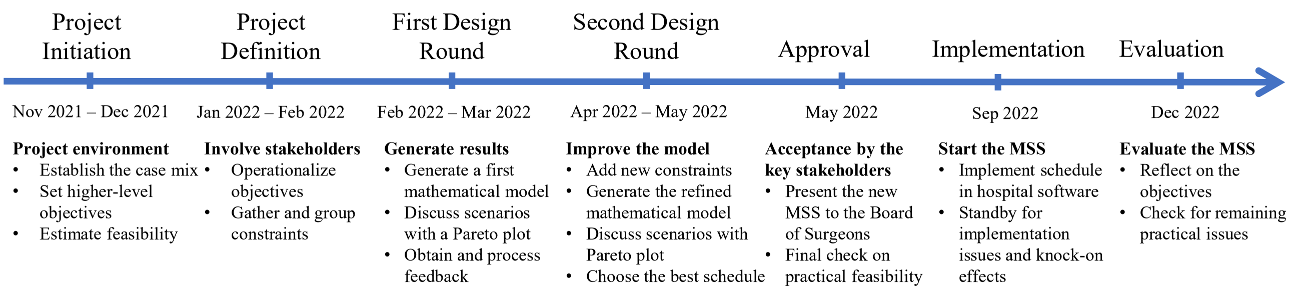

Project Initiation
Nov 2021 – Dec 2021
Project environment
• Establish the case mix
• Set higher-level objectives
• Estimate feasibility
Project Definition
Jan 2022 – Feb 2022
Involve stakeholders
• Operationalize objectives
• Gather and group constraints
First Design Round
Feb 2022 – Mar 2022
Generate results
• Generate a first mathematical model
• Discuss scenarios with a Pareto plot
• Obtain and process feedback
Second Design Round
Apr 2022 – May 2022
Improve the model
• Add new constraints
• Generate the refined mathematical model
• Discuss scenarios with Pareto plot
• Choose the best schedule
Approval
May 2022
Acceptance by the key stakeholders
• Present the new MSS to the Board of Surgeons
• Final check on practical feasibility
Implementation
Sep 2022
Start the MSS
• Implement schedule in hospital software
• Standby for implementation issues and knock-on effects
Evaluation
Dec 2022
Evaluate the MSS
• Reflect on the objectives
• Check for remaining practical issues

Figure 5.3: The project's timeline from initiation until implementation and evaluation.

- Scenario 1: The purchase of an additional liposuction machine, allowing more freedom in where plastic surgery could be placed in the schedule.
- Scenario 2: Limiting the number of laparoscopic kits available per day to 11.
- Scenario 3: Enforcing any split blocks for gynecology and bariatrics to be scheduled one after the other in the same operating room. This reduces the number of times in the week that additional surgical assistants need to be scheduled.
- Scenario 4: Permitting a particular surgeon to operate within the mastectomy cluster, which in fact *restricted* the days to which this cluster could be scheduled.
- Scenario 5: Limiting the bookings of gynecology, bariatrics and orthopedics to a maximum of 9 bookings per week (a booking is either a split block or a whole block). This provides consistency to the schedule of the surgical assistants.

The results were discussed with hospital management, clarifying the need and potential improvement of changing the schedule. This was soon clear to the management, who were delighted by the possible improvements shown. The new proposals additionally satisfy constraints not shown in the Pareto plot, such as dealing with the limited availability of C-bow X-radar machines for neurosurgery, urology, and trauma and the improved outflow to the oncology ward. The presentation for management included a table that showed which constraints were satisfied per schedule.

The costs for a new liposuction machine (scenario 1) were found to be insignificant when compared to the reduction of split surgery blocks and the improvement of outflow. Therefore, management decided to purchase a new liposuction machine that allowed cosmetic plastic surgery to be scheduled on Mondays. The new schedule '2nd liposuction machine' was sent out as a proposal to all surgeons, medical support staff and planners.

5.5.2 The Second Design Round

Within a few days after sending the new schedule, several emails arrived, claiming that the newly proposed schedule would be impossible to operate. The model thus required extra constraints:

- *Availability constraints*, a number of these were not communicated or interpreted correctly. E.g., vascular surgery could only operate a maximum of 3 afternoons and some plastic surgeons would be absent during certain weeks in each cycle
- *A bound on split blocks* was used for specific clusters with extra supporting personnel, e.g., bariatrics and gynecology use a third operating

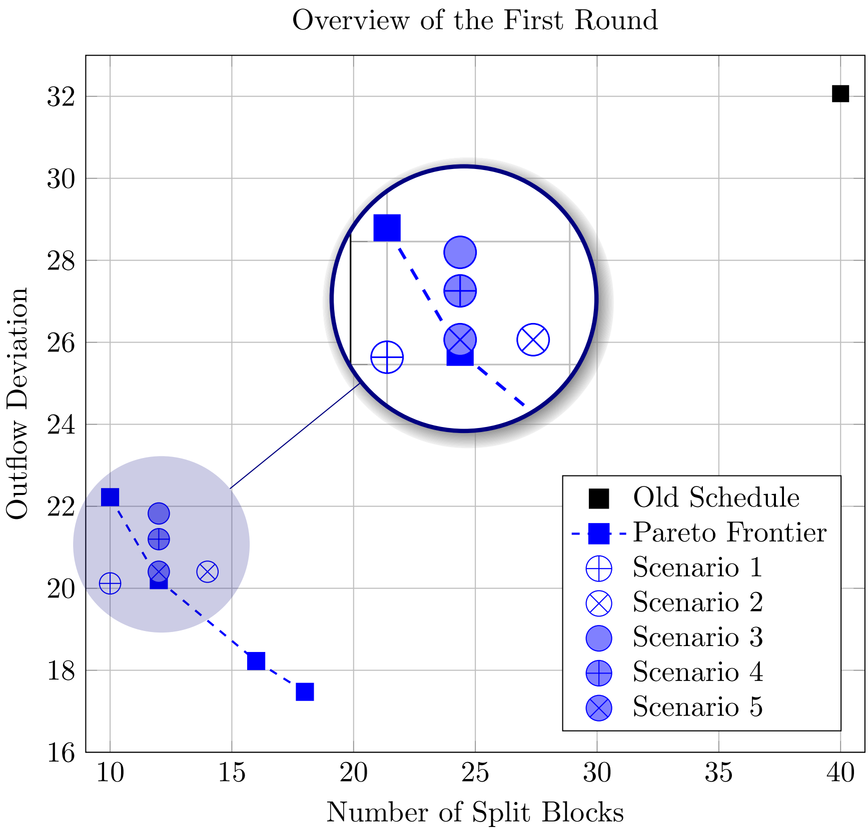


Figure 5.4: The Pareto frontier corresponds to the first round and is obtained by varying the weight on half sessions. Five additional scenarios are calculated to aid the decision-making process.

assistant due to the large number of scopic surgeries. In case of split blocks this leads to excess staff during the rest of the day.

- *Follow-up constraints* were introduced to avoid peak demand in the plaster room by preventing plastic surgeons from working subsequent days (inter-day). Other versions of this constraint avoided clusters of urology and gynecology following up on each other.
- *Rolling horizon constraints* were included for urology so that surgery time was spread equally over the schedule.

These, and other points, led to more than 25 additional demands being added to the model. With the newly added constraints, a new Pareto frontier was generated by varying the weight on split blocks and keeping the weights for clinical and daycare constant. The solutions corresponding to the second round are shown by the red dotted curve. As seen in Figure 5.5, the new constraints

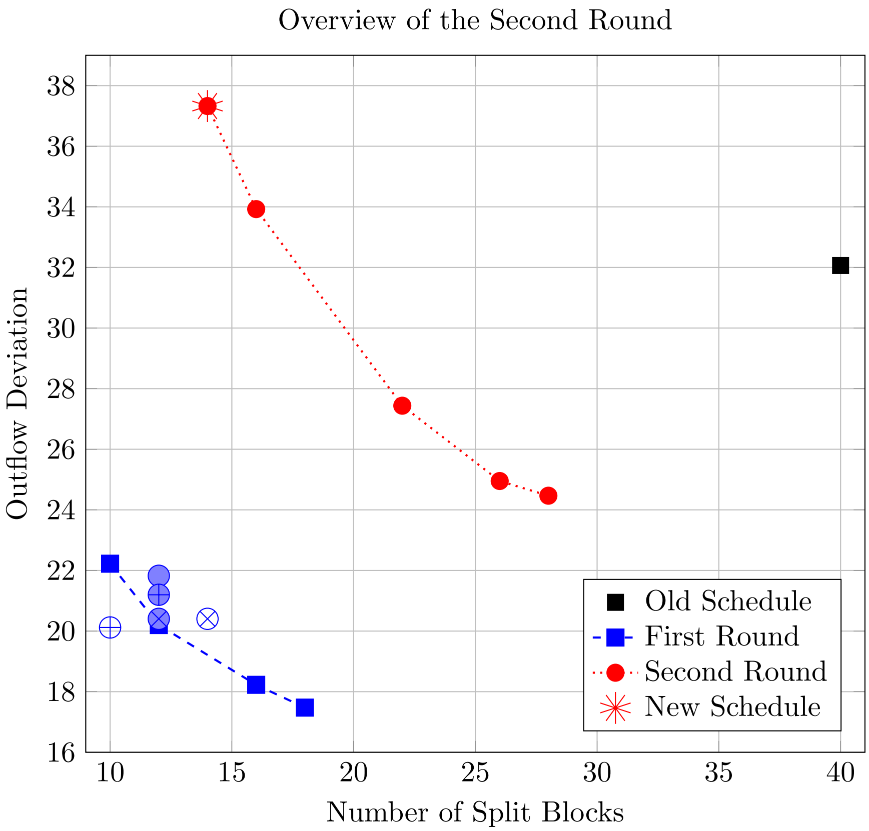


Figure 5.5: An overview with the Pareto frontiers of the two rounds; note that Figure 5.4 is encapsulated in this overview. The starred solution was chosen as the new schedule by the hospital's management.

restrict the number of possible solutions further, moving the frontier further outward, taking the number of features of the model to 2542 variables and 3560 constraints.

Showing the new results to the management of the hospital, it became clear that they were primarily concerned about hospital production, thereby preferring the schedule with just 14 split blocks. We brought to their attention that this solution would worsen the steadiness of the outflow to the wards, see Figure 5.6. Nevertheless, the outflow deviation was still considered acceptable, and improving it was subordinated to the goal of increasing production, which fewer split blocks would enable. As the next step following this meeting, the newly proposed schedule was sent out to all relevant stakeholders, asking whether the new schedule, presented in Table 5.3, could be put in operation. No new obstacles were identified and during the following meeting with the *Board of Surgeons* the schedule was considered for approval. They had some feedback: Orthopedics complained, but the nature of their complaint related purely to the reduction of OR time (case mix), not the schedule itself; Plastic surgery was not happy with the large number of changes, but agreed to implement the new schedule; and lastly, other surgical specialties expressed happiness to have found a suitable new schedule for the hospital.

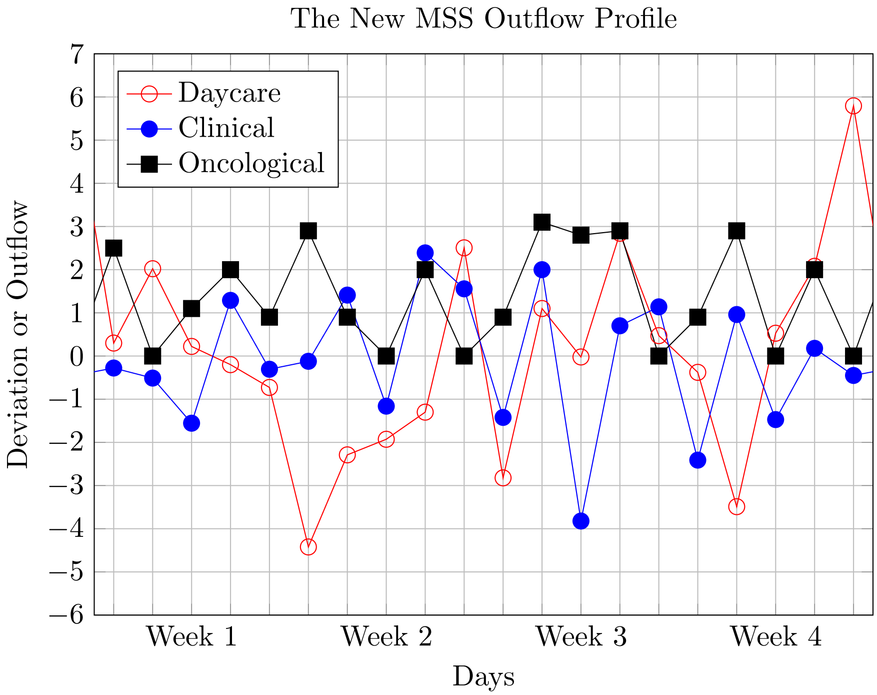


Figure 5.6: The corresponding outflow deviations from the target of the MSS for daycare and clinical as well as the net outflow to the oncological ward, which should ideally be well below 4.

5.5.3 Evaluation of the New MSS

Several significant differences can be spotted when comparing the new schedule, as shown in Table 5.3, with the old one, as shown in Table 5.2 (Section 5.3). One of the primary considerations reflected by the new schedule is the reduction of split blocks (gray blocks); it is reduced from 40 to 14—every four weeks—which was the minimum number of turnarounds while satisfying all the must-have constraints. This 65% reduction is beneficial for several reasons: it lowers the workload of supporting staff and frees up surgery time because turnarounds inherently take time off of the subsequent block. Also, valuable time is won on the morning blocks, because surgical cases are scheduled with a certain degree of conservatism not to overrun the session. Only considering the turnaround

		OR1	OR2	OR3	OR4	OR5	OR6
Week 1	Mon	URO URO*	ORT ORT	PRO PRO	BAR BAR	GEN GEN	COS COS
	Tue	TRA TRA	ORT ORT	ENT ENT	BAR BAR	MAS/PLA$_{Z}$ MAS/PLA$_{Z}$	PLA$_{M}$ PLA$_{M}^{*}$
	Wed		PLA$_{A}$ PLA$_{A}^{*}$	MAS MAS	BAR BAR		PLA$_{T}$ PLA$_{T}$
	Thu	URO URO	ORT ORT	ORG ORG	NEU$_{P}$ NEU$_{P}^{*}$	BRE BRE	PLA$_{L}$ PLA$_{L}$
	Fri	TRA TRA	ORT ORT	IHE VAS	GEN GEN	GYN GYN	PLA$_{T}$ PLA$_{T}^{*}$
Week 2	Mon	URO URO	ORT ORT	ENT ENT*	BAR BAR	GEN GEN	COS COS
	Tue	TRA TRA	MAS MAS*	ENT ENT	BAR BAR	PLA$_{Z}$ PLA$_{Z}$	PLA$_{T}$ PLA$_{T}$
	Wed		ORT ORT		BAR BAR	VAS GYN	PLA$_{M}$ PLA$_{M}^{*}$
	Thu	NEU$_{P}$ NEU$_{P}^{*}$	ORT ORT	ORG ORG	GEN GEN	MAS MAS	PLA$_{A}$ PLA$_{A}$
	Fri	VAS TRA	ORT ORT	PLA$_{W}$ PLA$_{W}$	BAR BAR	GYN GYN	PLA$_{L}$ PLA$_{L}^{*}$
Week 3	Mon	TRA TRA	ORT ORT	ENT ENT*	BAR BAR	GEN GEN	COS COS
	Tue	URO URO	ORT ORT	PLA$_{L}$ PLA$_{L}$	NEU$_{V}$ NEU$_{V}^{*}$	MAS MAS	PLA$_{Z}$ PLA$_{Z}$
	Wed			ENT ENT*	BAR BAR	MAS/PLA$_{I}$ MAS/PLA$_{I}$	PLA$_{T}$ PLA$_{T}$
	Thu	URO URO	ORT ORT	ORG ORG	GEN GEN	IHE IHE*	PLA$_{T}$ PLA$_{T}$
	Fri	TRA TRA	ORT ORT	CHC VAS	BAR BAR	GYN GYN	PLA$_{T}$ PLA$_{M}^{*}$
Week 4	Mon	PLA$_{T}$ PLA$_{T}^{*}$	ORT ORT	GEN GEN	BAR BAR	GYN GYN	COS COS
	Tue	URO URO	ORT ORT	PED TRA	MAS MAS	PLA$_{Z}$ PLA$_{Z}$	PLA$_{T}$ PLA$_{T}$
	Wed		ORT ORT	ENT ENT*	BAR BAR		CHC VAS
	Thu	GEN GEN	PLA$_{L}$ PLA$_{L}$	ENT ENT*	BAR BAR	MAS MAS	PLA$_{A}$ PLA$_{A}$
	Fri	TRA TRA	ORT ORT	IHE IHE*	VAS BAR	GYN GYN	PLA$_{W}$ PLA$_{W}$

Table 5.3: The new MSS, in which the asterisks ($*$) indicate which OR is dedicated to emergency surgeries.

time reduced improves annual production by 75 hours, which is estimated to increase surgery time by 0.6%—equivalent to surgery for 50 patients, amounting to 300 000 Euros.

For practitioners, it is important that their choices and corresponding improvements effectuate in practice when the new schedule is implemented. Therefore, the new MSS was evaluated after three months of implementation, December 2022. Users of the new schedule, among others surgeons, nurses and managers were asked to provide feedback. Summarizing the feedback, the following four points were distilled—furthermore, note that the typically positive notes are positive aspects that are not explicitly addressed, as is often the case with feedback.

1. *Increase in production.* Indeed, as projected, for the wards and surgery rooms, an increase in the production was felt and measured: 9.3% more patients when comparing the period October–November of 2022 to 2019. A number of measures are thought to have resulted in this increase, and the implementation of the new MSS was seen as an important one. The growth of 9.3% in production was a lot more than the expected 0.6%. A discussion on the positive effects of the MSS on the productivity of the hospital is given in Section 5.6.1.
2. *More early patient admissions.* The manager of the clinical ward noted an increase in early morning hospital admissions, which was confirmed with additional data analysis. This created a problem in capacity with the overlap of in-house patients and new intakes. The observation seemed at odds with the flexibility provided by the reduction in split blocks; whole blocks allow more options throughout the day to schedule patients requiring admission. However, after digging deeper, the underlying cause was simply the surge in production, which could not be offset by exploiting the flexibility of having more whole blocks.
3. *Heavy pressure on the recovery room.* For instance, surgery blocks with a high number of children led to peak occupation in the recovery room. While the model restricts the concurrency of pediatric blocks, no additional constraints were implemented to limit the occupation. However, there was no reason to further explore this issue.
4. *Missing a constraint.* For one of the specialties (ORG), a constraint that specified the availability was wrongly set to all Thursdays, which should have been for only three of the four. An issue that was solved internally without adjusting the schedule or optimization.

5.6 Conclusion and Discussion

We describe the creation and implementation of a new 4-week cyclic schedule that assigns blocks to (sub)specialties for a mid-sized Dutch hospital. The creation of this Master Surgery Schedule is particularly focused on practice and demonstrates how operations research can be successfully applied in healthcare. Specific objectives and a multitude of concerns are established in collaboration with the hospital, resulting in the inclusion of a large number of constraints regarding availability, concurrency, similarity, follow-up (inter- and intra-day) and rolling horizon, to ensure continuity of care. In the process, a mathematical optimization approach is used. The objective function established comprises tracking split blocks and outflow deviation by means of a tailored goal programming approach, taking costs and resource constraints into consideration. Note that minimizing turnarounds and outflow deviation are conflicting objectives, as the first one reduces the flexibility to accommodate a steadier outflow to wards.

In the entire development process, there was an active collaboration with the hospital managers and operations research specialists. This helped to identify which constraints were needed and which were of lesser importance. Also, it helped the direct involvement of stakeholders, which we believe has eased the adoption. To facilitate this, Pareto plots were generated to illustrate tradeoffs, and additional scenarios were considered to keep the hospital managers involved and in control. In the end, the schedule with the fewest number of turnarounds was selected in May 2022 to be implemented in September 2022, reducing the number of turnarounds by a staggering 65% (cf. Table 5.2) freed up surgery time, resulting in a monetary benefit of 300 000 Euros annually from extra production.

5.6.1 Assessment of the Implementation Process

Table 5.4 compares the old and new schedules for several indicators. The new plan did not obtain a better outflow to all wards—daycare remained the same while the outflow to the clinical ward worsened—because the reduction of turnarounds superseded the goal of a steadier outflow. It became apparent during the feedback of the design rounds that management steered toward reducing turnarounds, or equivalently number of split blocks.

Moreover, for modeling daycare outflow, one target value is chosen per day, and the model should approach this value as closely as possible. However, during the design rounds it became clear that daycare had enough physical space and that one nurse always cares for five patients. Therefore, an improvement of the model would have been to target any multiple of five for daycare. To encourage multiples of five the outflow targets were fine-tuned which allowed the daycare manager to schedule *one more* or *one fewer* shifts. This can be seen in Figure 5.7 where the dotted lines imply five alternative staffing occupations, as structurally applied for the new operating room schedule. Also, note that the outflow to the oncological ward does not have any violations, because the outflow targets were put in as a constraint.

According to the evaluation carried out by the hospital following the project, as detailed in Section 5.5.3, it was observed that there had been a

Consideration	MSS 2018	MSS 2022
Number of split blocks	40	14
Clinical outflow violation	21.1	25.1
Daycare outflow violation	35.0	35.4
Oncology outflow violations	2	0
Method	Manual (with simulation)	Mathematical optimization
Project workload	6 months (24 hours / week)	6 months (24 hours / week)
Schedules considered	16	Pareto frontier
Design rounds (until implementation)	4	2

Table 5.4: Comparison of the generation of the old (2018) and the new schedule (2022). The first three rows are the actual numbers of split blocks and outflow violations; these values were weighted in the objective function.

notable 9.3% increase in production. This observation serves to reinforce the idea that the reduction in turnaround time has contributed to an improvement in productivity, as likely has the reduction in time lost to conservative scheduling due to split blocks. This was not the only contributing factor, as there was a number of other changes that positively impacted productivity, some coincidental and some indirect. A coincidental factor, i.e., one not influenced by the new schedule, was the hospital management's decision to reduce emergency surgery time by 37.5% after the implementation. This freed up surgery time for regular surgery sessions, directly leading to an increase in production.

The *indirect benefits* from the new schedule are those that are not included in the objective function, but have had a positive impact. First, the new case mix gave an updated time division that is better aligned with the inflow of patients, leading to more blocks being filled. Second, updating availability for surgeons has decreased the chances of canceled procedures. Third, a number of new clusters have been formed for patients that were previously difficult to plan for, such as the combination blocks for mastectomy and plastic surgery, and clusters that target more specific patient groups, which has reduced changeover time between patients.

Besides the comparison on the performance dimensions, the processes by which a solution was found can also be compared. For the old schedule, a manual approach was used, with the help of simulation to check outflow and design-support tools to check the satisfaction of constraints. The capacity manager described that finding an appropriate schedule was a "*nightmare*" akin to "*solving an unsolvable Sudoku puzzle.*" A frustrating process as each time a schedule was computed and proposed new constraints would be brought up, rendering the proposed schedule unusable. In total, the iterative process resulted in the creation of 16 schedules over roughly four design rounds. The new schedule was developed via mathematical optimization supported by operations researchers and was felt to be "*far less frustrating*" due to the relative ease of

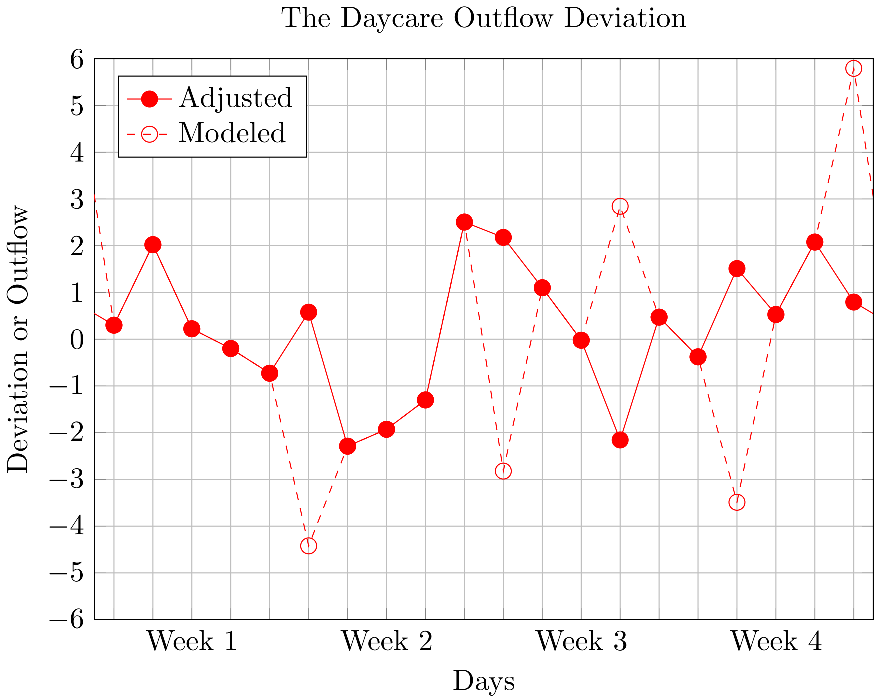


Figure 5.7: Revisting the outflow deviation to daycare.

including additional constraints. The challenge for the capacity manager was to collect the constraints as soon as possible—via design rounds—so that they could be included in the model. Due to these rounds, a clear reduction in the number of iterations was obtained, with the caveat that the (re)formulation of the mathematical model required additional time as the timelines were about the same.

5.6.2 Limitations of the Project

Some standard limitations apply as this work is a single case report; for example, it describes a successful implementation for a single Dutch hospital and how it compares to other or international hospitals. We, however, believe that the basis for each hospital is the same; only the political playing field, constraints, and targets will differ across hospitals. The last two can, of course, be put in a model.

Next, considering the modeling framework, an optimal solution for a mathematical program has some limitations, such as what is chosen as the objective and which modeling decisions are made. Also, reflecting on the feedback after the schedule, new challenges arise when a solution is implemented; as the feedback on the schedule in Section 5.5.3, handling more patients creates a new environment with possibly new problems. Furthermore, the development and input of the model are based on historical information, for example, about the case mix, outflow per block, and stakeholders' (working) requirements. These data are not guaranteed to be the same—in fact, that is highly unlikely. Moreover, besides variation in the data, the setting can change drastically, for example, emergency blocks were incorporated into the model but were shortened considerably immediately after implementation because of a management decree to increase production.

One more arguable shortcoming is that we forego explicit modeling of bed occupancy, which other applied works tend to do in much more detail. Variation in length of stay was not taken into consideration, although a few clusters did have longer average stays in the clinical ward. Modeling length of stay would have required greater granularity of the data and a higher level of complexity in the modeling (see van Essen et al. (2014), Adan et al. (2009), and Holte and Mannino (2013)).

Another avenue to deal with the uncertainty is to employ robust optimization techniques; for example, Bos et al. (2023) use it when designing a schedule with downstream capacity constraints. Also, one might wonder whether this detailed information cannot be better solved in scheduling surgeries at the operational level—so once the block schedule is set, see van den Broek d'Obrenan et al. (2020). For an integration of both, see, for example, Schneider et al. (2020), where they consider the operational decisions in the allocation of block scheduling.

5.6.3 Practical Recommendations

We have so far outlined the implementation of operations research in a challenging healthcare environment with high stakes and sizeable operational impact. Here, we want to provide some insight into why embracing operations research in healthcare decision-making is useful and how one should carry out such integration and implementation successfully. Reflecting on the trajectory of the project, we identify some success factors; some of these have already been mentioned in literature, while others appear to be novel.

Firstly, we consider for hospital management the benefits that operations research has to the hospital. As is mentioned by Blake and Donald (2002), the operations research model is seen as an objective decision maker, reducing the risk that the scheduler is used as a political tool to attain personal wishes — as did happen in the past, acknowledged by the hospital's capacity manager when drafting the old schedule in 2018. Second, also identified by Visintin et al. (2017), the model can be used to reveal preferences as was done with the Pareto plots in Section 5.5. This is useful not only for the operations researcher in developing the model, but can also help hospital management determine what trade-offs they find important. Third, as also mentioned by Blake and Donald (2002) and Visintin et al. (2017), once preferences have been determined, the model can be used to perform scenario analyses, which can depoliticize purchasing decisions, as with the question of the liposuction machine in Section 5.5.2

We will now consider recommendations for operations researchers by presenting as lessons learned four factors that we feel most helped enable the project. First, a sufficient degree of commitment or urgency is a prerequisite, also mentioned by Zenteno et al. (2016), who name commitment of high-level leadership and engagement of surgical services as essential. Second, a factor that we feel is very important, but do not see mentioned previously, is the inclusion of sufficient time to enable the full translation of practice into the terms of the model, which can be done by working in design rounds whereby stakeholder input is solicited. Third, at the same time one should be critical of stakeholder demands, discerning well between a true must-have constraint and a personal wish, a would-like constraint. Accepting too many constraints without researching the underlying motivation will drastically reduce the quality of the solution. This is encouraged by Van Oostrum et al. (2010) and is also mentioned by Visintin et al. (2017) as being an important factor in their case. Finally, a recommendation of ours is to be aware of knock-on effects, such as the requirement to redesign outpatient schedules. This could be handled by including them as constraints in the model immediately, but typically, they turn out to be *would-likes*, and the department could be asked to adapt.

There are also some additional success factors which complement the above. First, similar to Visintin et al. (2017), we suggest avoiding complex solution methods. Mixed integer linear optimization was the tool of choice in this project because there is a plethora of modeling languages and solvers available for MILPs which helps in swift development of the model. Second, this project benefited from the involvement of RKZ's capacity manager, who had knowledge of the inner workings of the hospital, held political sway, and could facilitate communication between stakeholders and the operations researchers. This echoes what can be seen in the work of Benchoff et al. (2017), who frequently refer to the interventions of the hospital's Associate Physician in Chief. We hope that these recommendations, together with the ones named above, can be leveraged by hospital management and operations research practitioners to stimulate the use of more operations research in healthcare practice.

5.7 The Mixed Integer Linear Program

The new MSS presented in the report was found by means of Mixed Integer Linear Optimization. The general set-up of the Mixed Integer Linear Program (MILP) is given below, after which we present a number of constraints that extended the MILP to better suit the particular problem, and which may be of interest to the applied researcher.

The MILP should minimize the number of split blocks while also balancing outflow. The solution should satisfy eclectic constraints often unique to each cluster. It should also ideally be able to be found quickly on a simple computer so that various scenarios can be investigated. An obvious approach might be to use decision variables x_{ijk} equal to 1 when cluster i is scheduled to block j in room k and then penalise split blocks directly, however this formulation introduces symmetries as the exact room k is often not important. Instead we implement a two-stage program that first assigns clusters to morning and afternoon blocks, penalising both the number of days on which a cluster is scheduled and the effect of this scheduling on outflow, and then in the second stage matches these morning and afternoon blocks to eliminate split blocks. The underlying logic is that if a cluster is assigned a morning and an afternoon block on the same day, a split block can be avoided.

In 5.7.1 we will give the baseline first stage sub-program. Constraints which extend the first stage sub-program can be found in Section 5.7.2. Finally, the second stage sub-program is presented in Section 5.7.3 together with an argument for optimality of the two-stage approach.

5.7.1 The Baseline MILP

The following is a standalone first stage MILP which can be expanded upon using select constraints given later. In the following $d \in D = \{0, 1, \dots, 19\}$ denotes days, while $j \in \{2d, 2d + 1 \,|\, d \in D\}$ denotes blocks. Note that there are 20 days and thus $j = 0, 1, \dots, 39$ blocks, with even j denoting morning blocks and odd j denoting afternoon blocks. $W = \{A, B\}$ denotes the clinical and daycare wards respectively. The main decision variable for this problem is $y_{ij} = 1$ when a cluster i is scheduled to block j and 0 otherwise. $r_{id} = 1$ when a cluster i is scheduled to the emergency reserved block and the variable $t_{id} = 1$ when a cluster is scheduled on a day is a book-keeping variable useful for formulating the objective function and some constraints. The variables δ (with indices omitted here for brevity) measure the degree of deviation between realised and target patient outflows for the purpose of goal programming.

The Mixed Integer Linear Program

$$\min \quad c_t \sum_{i\in I}\sum_{d\in D} t_{id} + c_A \sum_{d\in D}\sum_{\ell\in L_A} \delta^{\ell}_{d,A} + c_B \sum_{d\in D}\sum_{\ell\in L_B} \delta^{\ell}_{d,B} \tag{5.1}$$

subject to:

$$r_{id} \le 1 - \frac{y_{i,2d} + y_{i,2d+1}}{2} \qquad i \in I, d \in D \tag{5.2}$$

$$t_{id} \ge r_{id} + \frac{y_{i,2d} + y_{i,2d+1}}{2} \qquad i \in I, d \in D \tag{5.3}$$

$$\sum_{i\in I} r_{id} = 1 \qquad d \in D \tag{5.4}$$

$$\sum_{i\in I} y_{i,2d} + y_{i,2d+1} \le K_d \qquad d \in D \tag{5.5}$$

$$\sum_{d\in D} 4.5(y_{i,2d} + y_{i,2d+1}) + 7r_{id} \ge \mathrm{H}_i \qquad i \in I \tag{5.6}$$

$$\delta^{+}_{d,w} = \sum_{i} p_i^{d,w} - \mathrm{T}_{d,w} \qquad d \in D, w \in W \tag{5.7}$$

$$\delta^{-}_{d,w} = \mathrm{T}^{w}_{d} - \sum_{i} p_i^{d,w} \qquad d \in D, w \in W \tag{5.8}$$

$$\delta^{++}_{d,w} \ge \delta^{+}_{d,w} - 1 \qquad d \in D, w \in W \tag{5.9}$$

$$\delta^{--}_{d,w} \ge \delta^{-}_{d,w} - 1 \qquad d \in D, w = A \tag{5.10}$$

$$y_{ij} \in \{0,1\} \qquad i \in I, j \in J \tag{5.11}$$

$$r_{id} \in \{0,1\} \qquad i \in I, d \in D \tag{5.12}$$

$$t_{id} \in \{0,1\} \qquad i \in I, d \in D \tag{5.13}$$

$$\delta^{+}_{d,w}, \delta^{++}_{d,w}, \delta^{-}_{d,w} \ge 0 \qquad d \in D, w \in W \tag{5.14}$$

$$\delta^{--}_{d,w} \ge 0 \qquad d \in D, w = A \tag{5.15}$$

(5.1) Is the objective function, the first term of which penalises the number of days on which a cluster is scheduled and the second two terms of which penalise the mismatch in outflow to each ward, clinical and daycare (denoted A and B respectively); the notation of which is described in more detail below.

(5.2) Prevents a cluster from being scheduled to an emergency-reserved room and a split block on the same day.

(5.3) Enforces the variable t_{id} to be equal to 1 when a cluster i is scheduled on day d.

(5.5) Limits the number of blocks that can be scheduled to twice the number of free operating rooms.

(5.6) Ensures that each cluster is scheduled to a minimum number of hours per 4-week cycle. These correspond to case mix constraints.

(5.7) Describes for each ward, clinical and daycare, the extent to which outflow exceeds the target.

(5.8) Describes for each ward the extent to which outflow falls short of the target.

(5.9) Together with (5.14) defines for both wards variables δ_d^{++} which are positive when the *excess* of outflow is greater than one.

(5.10) Together with (5.15) defines (only for the clinical ward) variables δ_d^{--} which are positive only when the *deficit* of outflow is greater than one.

The variables $\delta_{d,B}^{\ell}$ and $\delta_{d,A}^{\ell}$ help define the goal programming component of the objective, measuring the degree of deviation between realized and target patient outflows. $\sum_i p_i^{d,w} - \mathrm{T}_{d,w}$ gives, as an example, the outflow to ward w in excess of the target $\mathrm{T}_{d,w}$ on day d, where $p_i^{d,w}$ is the number of patients discharged to ward w by cluster i on day d. The purpose of the levels $\ell \in L_A$ and L_B is to approximate in a piece-wise manner a quadratic cost function; one large deviation on a single day is considered worse than many smaller deviations on many days. The exception to this is sending too few patients to daycare, where a single large negative deviation was considered to be equivalent to many smaller negative deviations. This is as larger unused capacity at daycare can be filled with extra non-surgery blocks. This results in there being levels $\ell \in L_A := \{--, -, +, ++\}$ for clinical, but only $\ell \in L_B := \{-, +, ++\}$ for daycare. The cost functions for both clinical and daycare are depicted in Figure 5.8, where the expected deviation is given on the horizontal axis, and its corresponding cost on the vertical axis. As described in Section 5.4.1, a patient discharged to Clinical was determined to cost 3.3 times as much as a patient discharged to daycare, i.e., $c_A = 3.3$ and $c_B = 1$ in the objective (5.1).

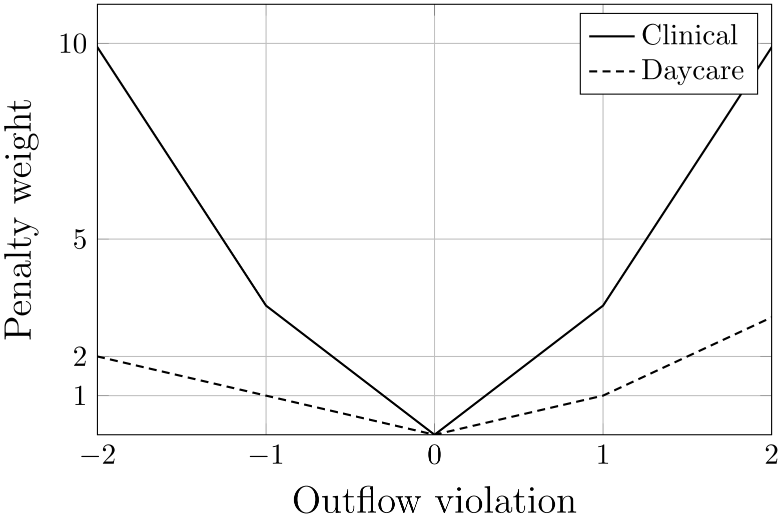


Figure 5.8: The piece-wise linear cost functions for outflow deviation used in the Mixed Integer Linear Program.

5.7.2 The Extended MILP

The following gives extensions to the standalone first stage MILP. Each extension comes with one or more examples. To aid in understanding the indices used in these examples the following tables are provided. Table 5.5 gives the days d numbered 0 through 19 for weeks 1 through 4, and Table 5.6 gives the indices j for the morning and afternoon blocks of each day.

	Mon	Tue	Wed	Thu	Fri
Week 1	0	1	2	3	4
Week 2	5	6	7	8	9
Week 3	10	11	12	13	14
Week 4	15	16	17	18	19

Table 5.5: The days $d \in D$. For example $d = 0, 5, 10, 15$ refers to every Monday throughout the 4 week cycle.

	Mon		Tue		Wed		Thu		Fri	
	M	A	M	A	M	A	M	A	M	A
Week 1	0	1	2	3	4	5	6	7	8	9
Week 2	10	11	12	13	14	15	16	17	18	19
Week 3	20	21	22	23	24	25	26	27	28	29
Week 4	30	31	32	33	34	35	36	37	38	39

Table 5.6: The blocks $j \in \{2d, 2d + 1 \,|\, d \in D\}$. For example $j = 0, 10, 20, 30$ refer to all Monday morning blocks.

Extension 1 (Concurrency constraints)**.** *The most common constraints in this problem were the concurrency constraints: preventing clusters from working simultaneously, and the availability constraints: restrictions on which clusters may work when. Concurrency constraints are modelled by:*

$$\sum_{i \in I'} y_{2d,i} + r_{d,i} \leq CC(d, I') \qquad d \in D, I' \subset I,$$
$$\sum_{i \in I'} y_{2d+1,i} + r_{d,i} \leq CC(d, I') \qquad d \in D, I' \subset I.$$

Example 1 (Concurrency constraints)**.** *On Tuesdays, Wednesdays and Thursdays, of Bariatrics (`BAR`), Gastroentorolgy (`GEN`) and Cholecystectomy (`CHC`) only two could operate:*

$$y_{2d,\texttt{BAR}} + y_{2d,\texttt{GEN}} + y_{2d,\texttt{CHC}} + r_{d,\texttt{BAR}} + r_{d,\texttt{GEN}} + r_{d,\texttt{CHC}} \leq 2, \ \textit{and}$$
$$y_{2d+1,\texttt{BAR}} + y_{2d+1,\texttt{GEN}} + y_{2d+1,\texttt{CHC}} + r_{d,\texttt{BAR}} + r_{d,\texttt{GEN}} + r_{d,\texttt{CHC}} \leq 2,$$
$$\textit{for } d \in \{\textit{Tue, Wed, Thu}\}.$$

Extension 2 (Availability constraints)**.** *Availability constraints are very simple and general, forbidding certain clusters from operating for certain blocks or on certain days:*

$$y_{2d,i} \ \textit{or} \ y_{2d+1,i} \ \textit{or} \ r_{d,i} \ \textit{or} \ t_{d,i} = 0 \qquad i \in I, d \in D.$$

They can be made more general by enforcing, for example $\sum_{d \in \widehat{D}} t_{i,d} \geq \kappa$ *for some subset of days* $\widehat{D} \subset D$*, as demonstrated in the second example that follows.*

Example 2 (Availability constraints)**.** *These constraints were used to prevent* Urology *from being scheduled to any slot on Tuesday or Thursday in Weeks 2 and 4 of the cycle:*

$$t_{6,\texttt{URO}} = t_{8,\texttt{URO}} = t_{16,\texttt{URO}} = t_{18,\texttt{URO}} = 0.$$

This can be made more general, for example Vascular Surgery (`VAS`) must be scheduled for a block on a minimum of two Wednesdays (days 2, 7, 12, and 17) and a block on a minimum of two Fridays (days 4, 9, 14, and 19):

$$t_{\texttt{VAS},2} + t_{\texttt{VAS},7} + t_{\texttt{VAS},12} + t_{\texttt{VAS},17} > 2,$$
$$t_{\texttt{VAS},4} + t_{\texttt{VAS},9} + t_{\texttt{VAS},14} + t_{\texttt{VAS},19} > 2.$$

Extension 3 (Further Availability Constraints)**.** *More complex availability constraints can be created from combinations of the decision variables. For example,* (5.16) *states that a cluster may never have a split block, that is if a cluster is scheduled for a morning block then it must also be scheduled for an afternoon block and vice versa;* (5.17) *states that a cluster may be scheduled to* *a morning block or a whole block, but never only an afternoon block; and* (5.18) *permits either a morning or an afternoon block, but forbids whole blocks:*

$$y_{i,2d} = y_{2d+1,i} \qquad i \in I',\, d \in D, \tag{5.16}$$

$$y_{i,2d+1} \leq y_{2d,i} \qquad i \in I',\, d \in D, \tag{5.17}$$

$$y_{i,2d} + y_{2d+1,i} \leq 1 \qquad i \in I',\, d \in D. \tag{5.18}$$

Example 3 (Further Availability Constraints)**.** *Mastectomy (`MAS`) was permitted only to operate whole blocks (constraint* (5.16)*); Ear-Nose-Throat (`ENT`) could operate a morning block or a whole block, but never an afternoon block in isolation (constraint* (5.17)*); Vascular Surgery (`VAS`) was only permitted to operate morning or afternoon blocks exclusively, that is no whole blocks (constraint* (5.18)*).*

Extension 4 (Intra-day Follow up constraints)**.** *Follow-up constraints prevent one cluster from being scheduled after another, either on the same day or across two days, due mainly to safety concerns or staffing levels. Intra-day follow up constraints are given by the following, stating that cluster i_1 given a morning block may not be followed by cluster i_2 on the same day, and vice versa:*

$$y_{i_1,2d} + y_{i_2,2d+1} \leq 1,$$
$$y_{i_2,2d} + y_{i_1,2d+1} \leq 1.$$

Example 4 (Intra-day follow up constraints)**.** *Urology and Gynecology were forbidden from following one another on any day to prevent the possibility that they may have to share an operating room. This can also be forbidden in the second stage by preventing a match between Urology and Gynecology split blocks, albeit it at the expense of additional split blocks:*

$$y_{\mathtt{GYN},2d} + y_{\mathtt{URO},2d+1} \leq 1 \quad \forall\, d \in D,$$
$$y_{\mathtt{URO},2d} + y_{\mathtt{GYN},2d+1} \leq 1 \quad \forall\, d \in D.$$

Extension 5 (Inter-day follow up constraints)**.** *Inter-day follow up constraints prevent one cluster from being scheduled the day after another for select days; the following forbids cluster i_1 from being scheduled to the day after i_2 and vice versa:*

$$t_{i_1,d} + t_{i_2,d+1} \leq 1,$$
$$t_{i_2,d} + t_{i_1,d+1} \leq 1.$$

Example 5 (Inter-day follow-up constraints)**.** *$\mathtt{PLA}_M$ was forbidden from following $\mathtt{PLA}_L$ and vice-versa on days Tuesday through Friday; this forbids the pairing of Monday and Tuesday, Tuesday and Wednesday, etc., but permits the pairing of Friday and the Monday of the next week:*

$$t_{\mathtt{PLA}_M,d} + t_{\mathtt{PLA}_L,d+1} \leq 1 \quad \textit{For all days except Fridays (days 4, 9, 14 \& 19),}$$
$$t_{\mathtt{PLA}_L,d} + t_{\mathtt{PLA}_M,d+1} \leq 1 \quad \textit{As above.}$$

Extension 6 (Similarity constraints). *The following constraints state that if a cluster from some group of clusters is scheduled to block j_1 then some cluster from that group must also be scheduled to a corresponding block j_2. The second line likewise enforces this for emergency reserved blocks:*

$$\sum_{i \in I'} y_{i,j_1} = \sum_{i \in I'} y_{i,j_2} \qquad I' \subset I, (j_1, j_2) \in \widehat{J} \times \widehat{J} \subset J \times J,$$
$$\sum_{i \in I'} r_{i,d_1} = \sum_{i \in I'} r_{i,d_2} \qquad I' \subset I, (d_1, d_2) \in \widehat{D} \times \widehat{D} \subset D \times D.$$

Example 6 (Similarity constraints). Mastectomy *was held to a two-week cyclical roster for its three component clusters,* `MAS`*,* `MAS/PLA`$_Z$*, and* `MAS/PLA`$_I$*, such that if one of its component clusters were scheduled to Monday morning (afternoon) in week 1, then one of its component clusters must also be scheduled to Monday morning (afternoon) in week 3. For compactness we write* `MASP`$_Z$ *and* `MASP`$_I$ *for* `MAS/PLA`$_Z$*, and* `MAS/PLA`$_I$ *respectively:*

$$y_{\mathtt{MAS},0} + y_{\mathtt{MASP}_Z,0} + y_{\mathtt{MASP}_I,0} = y_{\mathtt{MAS},20} + y_{\mathtt{MASP}_Z,20} + y_{\mathtt{MASP}_I,20},$$
$$y_{\mathtt{MAS},1} + y_{\mathtt{MASP}_Z,1} + y_{\mathtt{MASP}_I,1} = y_{\mathtt{MAS},21} + y_{\mathtt{MASP}_Z,21} + y_{\mathtt{MASP}_I,21}.$$

Extension 7 (Laprascopic kits). *The total number of laparoscopic kits used by all clusters operating on a given day I' must be less than some value Lap where lap_i^y is the expected number of laparoscopic kits used by cluster i during a morning or afternoon block and lap_i^r that used during an emergency reserved block:*

$$\sum_{i \in I'} lap_i^y \left(y_{2d,i} + y_{2d+1,i}\right) + lap_i^r r_{d,i} \leq Lap \qquad d \in D, I' \subset I. \tag{5.19}$$

Example 7 (Laparoscopic kits). *The clusters Inguinal Hernia (*`IHE`*), Bariatrics (*`BAR`*), Cholecystectomy (*`CHC`*), Gastroenterology (*`GEN`*), and Gynecology (*`GYN`*) should be scheduled such that their expected use of laparoscopic kits does not exceed 12 units on any given day. This limits which clusters may operate on the same day:*

$$\begin{aligned}
&4\left(y_{2d,\mathtt{IHE}} + y_{2d+1,\mathtt{IHE}}\right) + 6.22\, r_{d,\mathtt{IHE}}\\
&+ 3\left(y_{2d,\mathtt{BAR}} + y_{2d+1,\mathtt{BAR}}\right) + 4.67\, r_{d,\mathtt{BAR}}\\
&+ 3\left(y_{2d,\mathtt{CHC}} + y_{2d+1,\mathtt{CHC}}\right) + 4.67\, r_{d,\mathtt{CHC}}\\
&+ 2\left(y_{2d,\mathtt{GEN}} + y_{2d+1,\mathtt{GEN}}\right) + 3.11\, r_{d,\mathtt{GEN}}\\
&+ 1\left(y_{2d,\mathtt{GYN}} + y_{2d+1,\mathtt{GYN}}\right) + 1.56\, r_{d,\mathtt{GYN}} \leq 12\\
&\qquad\qquad\qquad\qquad\qquad\qquad \textit{for all } d \in D.
\end{aligned}$$

Extension 8 (Rolling horizon). *Rolling horizon constraints enforce upper and lower bounds on the number of hours a cluster may work in a set of days $\widehat{D}$.*

This permits the model flexibility in assigning (morning or afternoon) blocks. These constraints were also used extensively:

$$\sum_{d \in \widehat{D}} 4.5(y_{i,2d} + y_{i,2d+1}) + 7r_{id} \geq H_i(\widehat{D}, lo) \qquad i \in I, \widehat{D} \subset D,$$
$$\sum_{d \in \widehat{D}} 4.5(y_{i,2d} + y_{i,2d+1}) + 7r_{id} \leq H_i(\widehat{D}, hi) \qquad i \in I, \widehat{D} \subset D.$$

Example 8. *Urology (`URO`) was required to work between 9 and 18 hours across all Mondays, 18 and 27 hours across all Tuesdays, 18 and 27 hours across all Thursdays and – the example we depict – a minimum of 21.5 hours per fortnight. First consider the four possible fortnights in our schedule, numbered F_1 through F_4, where each fortnight is constructed from the days numbered* 0 *to* 19 *inclusive:*

$$\begin{aligned} F_1 &= \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}, \\ F_2 &= \{5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}, \\ F_3 &= \{10, 11, 12, 13, 14, 15, 16, 17, 18, 19\}, \\ F_4 &= \{15, 16, 17, 18, 19, 0, 1, 2, 3, 4\}. \end{aligned}$$

Then for each fortnight $F = F_1, F_2, F_3, F_4$ we enforce the following:

$$\sum_{d \in F} 4.5(y_{\texttt{URO},2d} + y_{\texttt{URO},2d+1}) + 7r_{\texttt{URO},d} \geq 21.5.$$

5.7.3 Splitting the Optimization

A two-stage approach was employed to avoid the problem of symmetry in the MILP. In this section we first provide the model, and then prove by contradiction that the two stage method indeed finds the optimal solution.

For a given day d let $I_d \subset I$ be the subset of clusters to be scheduled. Let K_d be the number of rooms available on that day, and define by $[K_d] := \{1, 2, \ldots, K_d\}$. Let $J := \{M, A\}$ be the morning and afternoon blocks. Let y_{ij} be one if cluster i is assigned to block j on this day. Let $x_{ijk} = 1$ if cluster i is scheduled to ‘room’ k in block j (the exact choice of room can be made later), and let s_i be the number of split blocks that room i is assigned on this day. The objective here is, once having assigned clusters to morning or afternoon blocks, to align those morning and afternoon blocks in order to minimize the number of split blocks in the schedule.

$$\min \sum_{i \in I_d} s_i \tag{5.20}$$

subject to:

$$\sum_{k \in [K_d]} x_{ikj} = y_{ij} \qquad i \in I_d, j \in \{M, A\} \tag{5.21}$$

$$\sum_{i \in I_d} x_{ikj} \leq 1 \qquad k \in [K_d], j \in J \tag{5.22}$$

$$s_i \geq x_{i,k,M} - x_{i,k,A} \qquad i \in I_d, k \in [K_d] \tag{5.23}$$

$$s_i \geq x_{i,k,A} - x_{i,k,M} \qquad i \in I_d, k \in [K_d] \tag{5.24}$$

$$x_{ikj} \in \{0, 1\} \qquad i \in I_d, k \in [K_d], j \in J \tag{5.25}$$

$$s_i \text{ free} \qquad i \in I_d \tag{5.26}$$

Note that we do not care about which cluster is scheduled to which room at this stage, only that clusters' morning and afternoon blocks are matched where possible.

We now address the issue of optimality, for which we provide a proof by contradiction. Suppose that the first and the second stage MILPs solve to optima, that is, stage 1 minimizes the number of days to which a cluster is placed, while stage 2 provides an optimal matching given a stage 1 solution. Call the schedule from stages 1 and 2 *Solution A*. Suppose that there is another *Solution B* with fewer split blocks than *Solution A*. There are two cases in which this can happen. In case 1, a cluster is scheduled to fewer days in Solution B than in Solution A, which violates the optimality assumption of Stage 1. In case 2, *Solutions A and B* designate each cluster the same number of days, but there are fewer mis-matched morning and afternoon blocks in *Solution B* than *Solution A*, violating the optimality assumption of Stage 2. Therefore, the two-stage approach minimizes the number of split blocks in the schedule. Furthermore, for a given minimum number of days to which a cluster is scheduled, the program in Stage 1 also minimizes the mismatch in outflow. Therefore the two-stage approach solves the complete problem to optimality.

5.8 Glossary of Cluster Names

Cluster	Abbreviation	Cluster	Abbreviation
Bariatrics	BAR	Orthopedic Surgery	ORT
Breast Reconstruction	BRE	Pediatric Surgery	PED
Cholecystectomy	CHC	Plastic Mastectomy Dr. I	$\mathrm{MAS/PLA}_{\mathrm{I}}$
Cosmetic Surgery	COS	Plastic Mastectomy Dr. Z	$\mathrm{MAS/PLA}_{\mathrm{Z}}$
Ear-Nose-Throat	ENT	Plastic Surgery Dr. A	$\mathrm{PLA}_{\mathrm{A}}$
Ear-Nose-Throat Dr. H	$\mathrm{ENT}_{\mathrm{H}}$	Plastic Surgery Dr. L	$\mathrm{PLA}_{\mathrm{L}}$
Ear-Nose-Throat Dr. K	$\mathrm{ENT}_{\mathrm{K}}$	Plastic Surgery Dr. M	$\mathrm{PLA}_{\mathrm{M}}$
Ear-Nose-Throat Dr. R	$\mathrm{ENT}_{\mathrm{R}}$	Plastic Surgery Dr. T	$\mathrm{PLA}_{\mathrm{T}}$
Gastroenterology	GEN	Plastic Surgery Dr. W	$\mathrm{PLA}_{\mathrm{W}}$
Gynecology	GYN	Plastic Surgery Dr. Z	$\mathrm{PLA}_{\mathrm{Z}}$
Inguinal Hernia	IHE	Proctoloy	PRO
Mastectomy	MAS	Trauma	TRA
Neurosurgery Dr. P	$\mathrm{NEU}_{\mathrm{P}}$	Urology	URO
Neurosurgery Dr. V	$\mathrm{NEU}_{\mathrm{V}}$	Vascular Surgery	VAS
Orthognathic Surgery	ORG		

Table 5.7: Glossary of cluster names and their abbreviations.

Chapter 6

Summary

The need for improving efficiency in healthcare is motivated largely by increasing global costs of healthcare. One possibility for improvement is optimization of the many schedules found within healthcare. This dissertation focuses on just that for two scheduling problems found within healthcare: the appointment scheduling problem and the master surgery scheduling problem. The basic appointment scheduling problem is a useful tool, but is limited in its scope: it does not allow for multiple resources (such as equipment, operating rooms, or doctors) or for patients with varying characteristics. The master surgery scheduling problem is well studied in theory, but its application, and thus reports on its application in literature, are lacking, especially outside of academic hospitals.

This dissertation aims to contribute to the literature on efficiency in healthcare by exploring generalizations of the appointment scheduling problem and applications of master surgery scheduling techniques in practice.

Methods and Results

In this dissertation we first consider an appointment schedule where the random distributions of service times differ between patients. This opens up the question of sequencing: the order in which patients of differing characteristics should be scheduled. To this end we develop a heuristic and apply it to two scheduling paradigms. This heuristic allows fast retrieval of solutions and motivates certain sequencing rules. The effectiveness of these sequencing rules is established, affirming the conclusion that the sequence in which patients arrive has at least as great an effect on the performance of a schedule as does the determination of their arrival times.

We also relax the *continuity of care* assumption by allowing patients to be seen by any one of multiple healthcare providers, termed *pooling.* This setting renders the *Lindley recursion* moot, and so *phase-type* distributions are chosen to model the system. These distributions suffer from a "curse of dimensionality", increasing rapidly in the number of *phases* required to describe the system, and becoming cumbersome as the number of patients and healthcare providers increases. Therefore, a simplifying technique is used that reduces the size of the problem while retaining the information required for optimization. A setting with multiple providers is difficult to analyze, and we can only conjecture that a unique optimum exists when the probability distributions of patients' service times are unimodal (it is worth mentioning that a counter-example exists for a bimodal distribution). Striking results are found both with regard to the shapes of the optimal solutions, which deviate from the typical *dome-shape*, and the impressive savings that can be had from pooling healthcare providers.

The model is extended to a *heavy-traffic* setting, where patients arrive closely together, providing analytical solutions that are simple to calculate and robust against mis-specification of the service time distribution.

This concept of pooling is expanded upon by having patients arrive in groups of two or more – so-called *batch arrivals*. This setting is intended to cover for a shortcoming in the previous case with pooling, where patients arrive one-by-one, for which individual physicians would have no personal appointment book, hindering planning and control of their day. As a jumping-off point we study this setting in its *steady state*, assuming *exponentially distributed* service times. We are able to show that the objective function of this particular formulation of a pooled system is convex, coming one step closer to answering the question of convexity conjectured above. We compare the performance of schedules with batch arrivals to that of schedules without, finding that expected waiting and idle times worsen, but only by small amounts. We also demonstrate how this problem can be studied in the transient setting, using a method that can later be expanded to other phase-type distributions beyond the exponential distribution.

Besides a theoretical contribution to appointment scheduling, we also look at the optimization of master surgery schedules in practice. We report upon the development and implementation of a master surgery schedule using linear optimization. We focus in particular on the process of collecting and implementing constraints, and managing stakeholder expectations. The schedule was optimized over a series of design rounds, at each step incorporating feedback from the hospital. The project was a success: it reduced the number of *split blocks*, cases where a surgical specialty operates for only half, and not a full day, from 40 to 14 per 4-week cycle, and satisfied a host of constraints, enabling an increase in production throughout the hospital. To aid in the implementation of similar projects elsewhere, we offer lessons learned from this project, including both factors that contributed to success and pitfalls that were encountered.

Recommendations

This dissertation considers the scheduling of patients both at primary and outpatient care – appointment scheduling – and of surgical specialties within a hospital – master surgery scheduling. From these studies we can arrive at some recommendations. Firstly, we suggest that when the data is available, the sequence in which patients are helped should be optimized, even if only heuristic methods are available. Secondly, we suggest that, whenever possible, multiple resources be pooled and that phase-type distributions are an excellent tool to be used in optimizing schedules in this case. Finally, we observe that well-established methods, such as linear optimization, are underutilized within healthcare and still have the potential to make a large difference. We strongly encourage both healthcare institutions to consider optimization and operations research practitioners to bring this possibility to their attention.

Bibliography

Adan, I., Bekkers, J., Dellaert, N., Vissers, J., and Yu, X. (2009). Patient mix optimisation and stochastic resource requirements: A case study in cardiothoracic surgery planning. *Health Care Management Science*, 12(2):129–141.

Adan, I., van Eenige, M., and Resing, J. (1995). Fitting discrete distributions on the first two moments. *Probability in the Engineering and Informational Sciences*, 9(4):623–632.

Ahmadi-Javid, A., Jalali, Z., and Klassen, K. (2017). Outpatient appointment systems in healthcare: A review of optimization studies. *European Journal of Operational Research*, 258(1):3–34.

Alvarez-Oh, H. J., Balasubramanian, H., Koker, E., and Muriel, A. (2018). Stochastic appointment scheduling in a team primary care practice with two flexible nurses and two dedicated providers. *Service Science*, 10(3):241–260.

Asmussen, S., Nerman, O., and Olssen, M. (1996). Fitting phase-type distributions via the EM algorithm. *Scandinavian Journal of Statistics*, 23(4):419–441.

Bailey, N. T. J. (1952). A study of queues and appointment systems in hospital out-patient departments, with special reference to waiting-times. *Journal of the Royal Statistical Society. Series B (Methodological)*, 14(2):185–199.

Balasubramanian, H., Banerjee, R., Denton, B., Naessens, J., and Stahl, J. (2010). Improving clinical access and continuity through physician panel redesign. *Journal of General Internal Medicine*, 25(10):1109–1115.

Bandi, C. and Gupta, D. (2020). Operating room staffing and scheduling. *Manufacturing & Service Operations Management*, 22(5):958–974.

Bar-Lev, S. K., Boxma, O., Perry, D., and Vastazos, L. P. (2017). Analysis and optimization of blood-testing procedures. *Probability in the Engineering and Informational Sciences*, 31(3):330–344.

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

Beliën, J., Demeulemeester, E., and Cardoen, B. (2009). A decision support system for cyclic master surgery scheduling with multiple objectives. *Journal of Scheduling*, 12(2):147.

Benchoff, B., Yano, C. A., and Newman, A. (2017). Kaiser permanente oakland medical center optimizes operating room block schedule for new hospital. *Interfaces*, 47(3):214–229.

Benjaafar, S. (1995). Performance bounds for the effectiveness of pooling in multi-processing systems. *European Journal of Operational Research*, 87(2):375–388.

Blake, J. T. and Carter, M. W. (2002). A goal programming approach to strategic resource allocation in acute care hospitals. *European Journal of Operational Research*, 140(3):541–561.

Blake, J. T. and Donald, J. (2002). Mount sinai hospital uses integer programming to allocate operating room time. *Interfaces*, 32(2):63–73.

Bos, H., Boucherie, R., Hans, E., and Leeftink, G. (2023). Distributionally robust scheduling of stochastic knapsack arrivals. *Available at SSRN 4385924*.

Bosch, P. M. V. and Dietz, D. C. (2001). Scheduling and sequencing arrivals to an appointment system. *Journal of Service Research*, 4(1):15–25.

Box, G. E. (1976). Science and statistics. *Journal of the American Statistical Association*, 71(356):791–799.

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

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

Çayırlı, T., Veral, E., and Rosen, H. (2006). Designing appointment scheduling systems for ambulatory care services. *Health Care Management Science*, 9(1):47–58.

Çayırlı, T., Veral, E., and Rosen, H. (2008). Assessment of patient classification in appointment system design. *Production and Operations Management*, 17(3):338–353.

Çayırlı, T. and Yang, K. K. (2014). A universal appointment rule with patient classification for service times, no-shows, and walk-ins. *Service Science*, 6(4):274–295.

Çayırlı, T., Yang, K. K., and Quek, S. A. (2012). A universal appointment rule in the presence of no-shows and walk-ins. *Production and Operations Management*, 21(4):682–697.

Chaudhry, M. L. and Kim, J. J. (2016). Analytically elegant and computationally efficient results in terms of roots for the GI^{X}/M/c queueing system. *Queueing Systems*, 82(1-2):237–257.

Childers, C. P. and Maggard-Gibbons, M. (2018). Understanding costs of care in the operating room. *JAMA surgery*, 153(4):e176233–e176233.

Choi, S. and Wilhelm, W. E. (2020). Sequencing in an appointment system with deterministic arrivals and non-identical exponential service times. *Computers & Operations Research*, 117:104901.

Collar, R. M., Shuman, A. G., Feiner, S., McGonegal, A. K., Heidel, N., and Duck, M. e. a. (2012). Lean management in academic surgery. *Journal of the American College of Surgeons*, 214(6):928–936.

Côté, M. and Stein, W. (2007). A stochastic model for a visit to the doctor's office. *Mathematical and Computer Modelling*, 45(3–4):309–323.

Coughlan, P. and Coghlan, D. (2002). Action research for operations management. *International Journal of Operations & Production Management*, 22(2):220–240.

Cox, T., Birchall, J., and Wong, H. (1985). Optimising the queuing system for an ear, nose and throat outpatient clinic. *Journal of Applied Statistics*, 12(2):113–126.

Daley, D. J. (1998). Some results for the mean waiting-time and workload in GI/GI/k queues. In *Frontiers in Queueing: Models and Applications in Science and Engineering*, pages 35–59. CRC Press.

De Kemp, M. A., Mandjes, M., and Olver, N. (2021). Performance of the smallest-variance-first rule in appointment sequencing. *Operations Research*, 69(6):1909–1935.

De Kok, A. (1989). A moment-iteration method for approximating the waiting-time characteristics of the GI/G/1 queue. *Probability in the Engineering and Informational Sciences*, 3(2):273–287.

De Koning, H., Verver, J. P., van den Heuvel, J., Bisgaard, S., and Does, R. J. (2006). Lean six sigma in healthcare. *Journal for Healthcare Quality*, 28(2):4–11.

De Vuyst, S., Bruneel, H., and Fiems, D. (2014). Computationally efficient evaluation of appointment schedules in health care. *European Journal of Operational Research*, 237(3):1142–1154.

Delesie, L. (1998). Bridging the gap between clinicians and health managers. *European Journal of Operational Research*, 105(2):248–256.

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

Denton, B., Viapiano, J., and Vogl, A. (2007). Optimization of surgery sequencing and scheduling decisions under uncertainty. *Health Care Management Science*, 10(1):13–24.

Denton, B. T., Miller, A. J., Balasubramanian, H. J., and Huschka, T. R. (2010a). Optimal allocation of surgery blocks to operating rooms under uncertainty. *Operations Research*, 58(4–1):802–816.

Denton, B. T., Miller, A. J., Balasubramanian, H. J., and Huschka, T. R. (2010b). Optimal allocation of surgery blocks to operating rooms under uncertainty. *Operations Research*, 58(4):802–816.

Dexter, F. and Epstein, R. H. (2005). Operating room efficiency and scheduling. *Current Opinion in Anesthesiology*, 18(2):195–198.

Dexter, F., Epstein, R. H., and Schwenk, E. S. (2019). Tardiness of starts of surgical cases is not substantively greater when the preceding surgeon in an operating room is of a different versus the same specialty. *Journal of Clinical Anesthesia*, 53:20–26.

El-Sharo, M., Zheng, B., Yoon, S. W., and Khasawneh, M. T. (2015). An overbooking scheduling model for outpatient appointments in a multi-provider clinic. *Operations Research for Health Care*, 6:1–10.

Fazal, F., Saleem, T., Rehman, M. E. U., Haider, T., Khalid, A. R., Tanveer, U., Mustafa, H., Tanveer, J., and Noor, A. (2022). The rising cost of healthcare and its contribution to the worsening disease burden in developing countries. *Annals of Medicine and Surgery*, 82.

Fei, H., Meskens, N., and Chu, C. (2010). A planning and scheduling problem for an operating theatre using an open scheduling strategy. *Computers & Industrial Engineering*, 58(2):221–230.

Fenton, L. (1960). The sum of log-normal probability distributions in scatter transmission systems. *IRE Transactions on Communications Systems*, 8(1):57–67.

Fries, B. E. and Marathe, V. P. (1981). Determination of optimal variable-sized multiple-block appointment systems. *Operations Research*, 29(2):324–345.

Gallego, G. and Moon, I. (1993). The distribution free newsboy problem: review and extensions. *Journal of the Operational Research Society*, 44(8):825–834.

Gao, Y., Zhang, Q., Lau, C. K., and Ram, B. (2022). Robust appointment scheduling in healthcare. *Mathematics*, 10(22):4317.

Gontijo, G., Atuncar, G., Cruz, F., and Kerbache, L. (2011). Performance evaluation and dimensioning of $\mathrm{GI}^{\mathrm{X}}/\mathrm{M}/\mathrm{c}/\mathrm{N}$ systems through kernel estimation. *Mathematical Problems in Engineering*, 2011.

Grassmann, W. K. (1988). Finding the right number of servers in real-world queuing systems. *Interfaces*, 18(2):94–104.

Green, L. V., Savin, S., and Lu, Y. (2013). Primary care physician shortages could be eliminated through use of teams, nonphysicians, and electronic communication. *Health Affairs*, 32(1):11–19.

Guda, H., Dawande, M., Janakiraman, G., and Jung, K. S. (2016). Optimal policy for a stochastic scheduling problem with applications to surgical scheduling. *Production and Operations Management*, 25(7):1194–1202.

Guido, R. and Conforti, D. (2017). A hybrid genetic approach for solving an integrated multi-objective operating room planning and scheduling problem. *Computers & Operations Research*, 87:270–282.

Gul, S., Denton, B. T., Fowler, J. W., and Huschka, T. (2011). Bi-criteria scheduling of surgical services for an outpatient procedure center. *Production and Operations Management*, 20(3):406–417.

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

Gupta, D. and Denton, B. (2008). Appointment scheduling in health care: Challenges and opportunities. *IIE Transactions*, 40(9):800–819.

Hanly, M. J., Churches, T., Fitzgerald, O., Caterson, I., MacIntyre, C. R., and

Jorm, L. (2021). Modelling vaccination capacity at mass vaccination hubs and general practice clinics. *medRxiv*.

Harders, M., Malangoni, M. A., Weight, S., and Sidhu, T. (2006). Improving operating room efficiency through process redesign. *Surgery*, 140(4):509–516.

Harel, A. (1990). Convexity results for single-server queues and for multi-server queues with constant service times. *Journal of Applied Probability*, 27(2):465–468.

Harper, P. R. and Gamlin, H. (2003). Reduced outpatient waiting times with improved appointment scheduling: a simulation modelling approach. *OR Spectrum*, 25(2):207–222.

Hassin, R. and Mendel, S. (2008). Scheduling arrivals to queues: a single-server model with no-shows. *Management Science*, 54(3):565–572.

Ho, C. J. and Lau, H. S. (1992). Minimizing total cost in scheduling outpatient appointments. *Management Science*, 38(12):1750–1764.

Holte, M. and Mannino, C. (2013). The implementor/adversary algorithm for the cyclic and robust scheduling problem in health-care. *European Journal of Operational Research*, 226(3):551–559.

Hulshof, P. J., Kortbeek, N., Boucherie, R. J., Hans, E. W., and Bakker, P. J. (2012). Taxonomic classification of planning decisions in health care: A structured review of the state of the art in OR/MS. *Health Systems*, 1(2):129–175.

Jafarnia-Jahromi, M. and Jain, R. (2020). Non-indexability of the stochastic appointment scheduling problem. *Automatica*, 118:109016.

Jansson, B. (1966). Choosing a good appointment system—a study of queues of the type (D, M, 1). *Operations Research*, 14(2):292–312.

Jeurissen, P., Maarse, H., et al. (2021). *The market reform in Dutch health care: Results, lessons and prospects.*

Jung, K. S., Pinedo, M., Sriskandarajah, C., and Tiwari, V. (2019). Scheduling elective surgeries with emergency patients at shared operating rooms. *Production and Operations Management*, 28(6):1407–1430.

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

Kemper, B., Klaassen, C., and Mandjes, M. (2014). Optimized appointment scheduling. *European Journal of Operational Research*, 239(1):243–255.

Kendall, D. G. (1953). Stochastic Processes Occurring in the Theory of Queues and their Analysis by the Method of the Imbedded Markov Chain. *The Annals of Mathematical Statistics*, 24(3):338–354.

Kiefer, J. and Wolfowitz, J. (1955). On the theory of queues with many servers. *Transactions of the American Mathematical Society*, 78(1):1–18.

Kim, C. S., Spahlinger, D. A., Kin, J. M., and Billi, J. E. (2006). Lean health care: what can hospitals learn from a world-class automaker? *Journal* *of Hospital Medicine: An official publication of the Society of Hospital Medicine*, 1(3):191–199.

Kim, S., Pasupathy, R., and Henderson, S. G. (2015). A guide to sample average approximation. *Handbook of Simulation Optimization*, pages 207–243.

Klassen, K. and Rohleder, T. (1996). Scheduling outpatient appointments in a dynamic environment. *Journal of Operations Management*, 14(2):83–101.

Klassen, K. J. and Yoogalingam, R. (2009). Improving performance in outpatient appointment services with a simulation optimization approach. *Production and Operations Management*, 18(4):447–458.

Klassen, K. J. and Yoogalingam, R. (2014). Strategies for appointment policy design with patient unpunctuality. *Decision Sciences*, 45(5):881–911.

Klassen, K. J. and Yoogalingam, R. (2019). Appointment scheduling in multi-stage outpatient clinics. *Health Care Management Science*, 22(2):229–244.

Kleiber, C. and Kotz, S. (2003). *Statistical size distributions in economics and actuarial sciences*. John Wiley & Sons.

Köllerström, J. (1974). Heavy traffic theory for queues with several servers. i. *Journal of Applied Probability*, 11(3):544–552.

Kong, Q., Lee, C. Y., Teo, C.-P., and Zheng, Z. (2016). Appointment sequencing: Why the smallest-variance-first rule may not be optimal. *European Journal of Operational Research*, 255(3):809–821.

Kortbeek, N., Braaksma, A., Burger, C. A., Bakker, P. J., and Boucherie, R. J. (2015). Flexible nurse staffing based on hourly bed census predictions. *International Journal of Production Economics*, 161:167–180.

Kuiper, A., Kemper, B., and Mandjes, M. (2015). A computational approach to optimized appointment scheduling. *Queueing Systems*, 79(1):5–36.

Kuiper, A. and Lee, R. H. (2022). Appointment scheduling for multiple servers. *Management Science*, 68(10):7422–7440.

Kuiper, A., Lee, R. H., van Ham, V. J., and Does, R. J. (2022). A reconsideration of lean six sigma in healthcare after the covid-19 crisis. *International Journal of Lean Six Sigma*, 13(1):101–117.

Kuiper, A. and Mandjes, M. (2015). Appointment scheduling in tandem-type service systems. *Omega*, 57:145–156.

Kuiper, A., Mandjes, M., and de Mast, J. (2017). Optimal stationary appointment schedules. *Operations Research Letters*, 45(6):549–555.

Kuiper, A., Mandjes, M., de Mast, J., and Brokkelkamp, R. (2023). A flexible and optimal approach for appointment scheduling in healthcare. *Decision Sciences*, 54(1):85–100.

Laxmi, P. V. and Gupta, U. (2000). Analysis of finite-buffer multi-server queues with group arrivals: $\mathrm{GI}^{\mathrm{X}}/\mathrm{M}/\mathrm{c}$. *Queueing Systems*, 36(1):125–140.

Lee, R. H. and Kuiper, A. (2024). Optimal sequencing using a scheduling heuristic. *Computers & Operations Research*, 161.

Lindley, D. (1952). The theory of queues with a single server. *Mathematical Proceedings of the Cambridge Philosophical Society*, 48(2):277–289.

Liu, L. and Liu, X. (1998). Block appointment systems for outpatient clinics with multiple doctors. *Journal of the Operational Research Society*, 49(12):1254–1259.

Main, O. (1995). What makes a well-oiled scheduling system? *OR Manager*.

Mak, H. Y., Rong, Y., and Zhang, J. (2014). Sequencing appointments for service systems using inventory approximations. *Manufacturing & Service Operations Management*, 16(2):251–262.

Mak, H.-Y., Rong, Y., and Zhang, J. (2015). Appointment scheduling with limited distributional information. *Management Science*, 61(2):316–334.

Mancilla, C. and Storer, R. (2012). A sample average approximation approach to stochastic appointment sequencing and scheduling. *IIE Transactions*, 44(8):655–670.

Mandelbaum, A., Momčilović, P., Trichakis, N., Kadish, S., Leib, R., and Bunnell, C. A. (2020). Data-driven appointment-scheduling under uncertainty: The case of an infusion unit in a cancer center. *Management Science*, 66(1):243–270.

Marques, I., Captivo, M. E., and Barros, N. (2019). Optimizing the master surgery schedule in a private hospital. *Operations Research for Health Care*, 20:11–24.

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

May, J. H., Strum, D. P., and Vargas, L. G. (2000). Fitting the lognormal distribution to surgical procedure times. *Decision Sciences*, 31(1):129–148.

Millhiser, W. P. and Veral, E. A. (2015). Designing appointment system templates with operational performance targets. *IIE Transactions on Healthcare Systems Engineering*, 5(3):125–146.

Ministry of Health, W. and Sport (2022). Integraal Zorgakkoord: 'Samen werken aan gezonde zorg' (dutch). [Online; accessed 30-August-2023].

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

Neuts, M. (1981). *Matrix-geometric solutions in stochastic models: An algorithmic approach*. Johns Hopkins University Press.

NHS (2022). The healthcare ecosystem. `https://www.who.int/data/gho/data/indicators/indicator-details/GHO/current-health-expenditure-(che)`

-as-percentage-of-gross-domestic-product-(gdp)-(-). (Accessed 27-Sep-2023).

Phillips, R. L. (2005). Primary care in the united states: problems and possibilities. *British Medical Journal*, 331(7529):1400–1402.

Radnor, Z. J., Holweg, M., and Waring, J. (2012). Lean in healthcare: the unfilled promise? *Social Science & Medicine*, 74(3):364–371.

Ridder, A., Van Der Laan, E., and Salomon, M. (1998). How larger demand variability may lead to lower costs in the newsvendor problem. *Operations Research*, 46(6):934–936.

Rising, E., Baron, R., and Averill, B. (1973). A systems analysis of a university-health-service outpatient clinic. *Operations Research*, 21(5):1030–1047.

Robinson, L. W. and Chen, R. R. (2011). Estimating the implied value of the customer's waiting time. *Manufacturing & Service Operations Management*, 13(1):53–57.

Rohleder, T. and Klassen, K. (2002). Rolling horizon appointment scheduling: A simulation study. *Health Care Management Science*, 5(3):201 – 209.

Salzarulo, P. A., Bretthauer, K. M., Côté, M. J., and Schultz, K. L. (2011). The impact of variability and patient information on health care system performance. *Production and Operations Management*, 20(6):848–859.

Samudra, M., Van Riet, C., Demeulemeester, E., Cardoen, B., Vansteenkiste, N., and Rademakers, F. E. (2016). Scheduling operating rooms: Achievements, challenges and pitfalls. *Journal of Scheduling*, 19:493–525.

Sang, P., Begen, M. A., and Cao, J. (2021). Appointment scheduling with a quantile objective. *Computers & Operations Research*, 132:105295.

Saremi, A., Jula, P., ElMekkawy, T., and Wang, G. G. (2013). Appointment scheduling of outpatient surgical services in a multistage operating room department. *International Journal of Production Economics*, 141(2):646–658.

Schäfer, W., Kroneman, M., Boerma, W., Van den Berg, M., Westert, G., Devillé, W., Van Ginneken, E., Organization, W. H., et al. (2010). The Netherlands: health system review. *World Health Organization. Regional Office for Europe.*

Schneider, A. T., van Essen, J. T., Carlier, M., and Hans, E. W. (2020). Scheduling surgery groups considering multiple downstream resources. *European Journal of Operational Research*, 282(2):741–752.

Schulz, A. S. and Udwani, R. (2019). Robust appointment scheduling with heterogeneous costs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Shaked, M. and Shanthikumar, J. G. (1990). Parametric stochastic convexity and concavity of stochastic processes. *Annals of the Institute of Statistical Mathematics*, 42(3):509–531.

Shanthikumar, J. G. and Yao, D. D. (1991). Strong stochastic convexity: Closure properties and applications. *Journal of Applied Probability*, 28(1):131–145.

Shunko, M., Niederhoff, J., and Rosokha, Y. (2018). Humans are not machines: The behavioral impact of queueing design on service time. *Management Science*, 64(1):453–473.

Sickinger, S. and Kolisch, R. (2009). The performance of a generalized bailey–welch rule for outpatient appointment scheduling under inpatient and emergency demand. *Health Care Management Science*, 12(4):408–419.

Soltani, M., Samorani, M., and Kolfal, B. (2019). Appointment scheduling with multiple providers and stochastic service times. *European Journal of Operational Research*, 277(2):667–683.

Song, H., Tucker, A. L., and Murrell, K. L. (2015). The diseconomies of queue pooling: An empirical investigation of emergency department length of stay. *Management Science*, 61(12):3032–3053.

Soriano, A. (1966). Comparison of two scheduling systems. *Operations Research*, 14(3):388–397.

Soroush, H. (1999). Sequencing and due-date determination in the stochastic single machine problem with earliness and tardiness costs. *European Journal of Operational Research*, 113(2):450–468.

Srinivas, S. and Choi, S. S. (2022). Designing variable-sized block appointment system under time-varying no-shows. *Computers & Industrial Engineering*, 172:108596.

Stein, W. E. and Côté, M. J. (1994). Scheduling arrivals to a queue. *Computers & Operations Research*, 21(6):607–614.

Sun, B., Evans, G. W., and Bai, L. (2011). Simulation modeling and analysis of a multi-resource medical clinic. In *2011 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, pages 593–600.

Swisher, J., Jacobson, S., Jun, J., and Balci, O. (2001). Modeling and analyzing a physician clinic environment using discrete-event (visual) simulation. *Computers & Operations Research*, 28(2):105–125.

Tijms, H. (1986). *Stochastic Modelling and Analysis — a Computational Approach.* Wiley Series in Probability and Mathematical Statistics: Applied Probability and Statistics. John Wiley & Sons, Chichester, UK.

Tijms, H. C. (2003). *A First Course in Stochastic Models.* John Wiley & Sons.

Van Buuren, M., Jagtenberg, C., Van Barneveld, T., Van Der Mei, R., and Bhulai, S. (2018). Ambulance dispatch center pilots proactive relocation policies to enhance effectiveness. *Interfaces*, 48(3):235–246.

van den Broek d'Obrenan, A., Ridder, A., Roubos, D., and Stougie, L. (2020). Minimizing bed occupancy variance by scheduling patients under uncertainty. *European Journal of Operational Research*, 286(1):336–349.

van Dijk, N. M. and van der Sluis, E. (2008). To pool or not to pool in call centers. *Production and Operations Management*, 17(3):296–305.

van Essen, J. T., Bosch, J. M., Hans, E. W., van Houdenhoven, M., and Hurink, J. L. (2014). Reducing the number of required beds by rearranging the OR-schedule. *OR Spectrum*, 36:585–605.

van Ham, V., Lee, R. H., and Kuiper, A. (2023). Optimizing and implementing a new master surgery schedule: Increasing productivity and balancing outflow. *Available at SSRN: abstract id 4634128*.

Van Oostrum, J. M., Bredenhoff, E., and Hans, E. W. (2010). Suitability and managerial implications of a master surgical scheduling approach. *Annals of Operations Research*, 178:91–104.

Vanberkel, P. T., Boucherie, R. J., Hans, E. W., Hurink, J. L., Van Lent, W. A., and Van Harten, W. H. (2011). Accounting for inpatient wards when developing master surgical schedules. *Anesthesia & Analgesia*, 112(6):1472–1479.

Vanden Bosch, P. M. and Dietz, D. C. (2000). Minimizing expected waiting in a medical appointment system. *IIE Transactions*, 32(9):841–848.

Vink, W., Kuiper, A., Kemper, B., and Bhulai, S. (2015). Optimal appointment scheduling in continuous time: The lag order approximation method. *European Journal of Operational Research*, 240(1):213–219.

Visintin, F., Cappanera, P., Banditori, C., and Danese, P. (2017). Development and implementation of an operating room scheduling tool: An action research study. *Production Planning & Control*, 28(9):758–775.

Wang, L., Demeulemeester, E., Vansteenkiste, N., and Rademakers, F. E. (2021). Operating room planning and scheduling for outpatients and inpatients: A review and future research. *Operations Research for Health Care*, 31:100323.

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

Wang, P. P. (1997). Optimally scheduling n customer arrival times for a single-server system. *Computers & Operations Research*, 24(8):703–716.

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

Weber, R. R. (1980). Note on the marginal benefit of adding servers to G/GI/m queues. *Management Science*, 26(9):946–951.

Weiss, E. (1990). Models for determining estimated start times and case orderings in hospital operating rooms. *IIE Transactions*, 22(2):143–150.

Welch, J. D. and Bailey, N. T. J. (1952). Appointment systems in hospital outpatient departments. *The Lancet*, 259(6718):1105–1108.

Wemelsfelder, M. L., den Hertog, D., Wisman, O., Ihalainen, J., and Janssen, M. P. (2022). Determining optimal locations for blood distribution centers. *Transfusion*, 62(12):2515–2524.

White, D. L., Froehle, C. M., and Klassen, K. J. (2011). The effect of integrated scheduling and capacity policies on clinical efficiency. *Production and Operations Management*, 20(3):442–455.

Whitt, W. (1982). Existence of limiting distributions in the GI/G/s queue. *Mathematics of Operations Research*, 7(1):88–94.

Whitt, W. (1983). Comparison conjectures about the M/G/s queue. *Operations Research Letters*, 2(5):203–209.

World Health Organization (2023). Current health expenditure (che) as percentage of gross domestic product (gdp) (%). `https://www.who.int/data/gho/data/indicators/indicator-details/GHO/current-health-expenditure-(che)-as-percentage-of-gross-domestic-product-(gdp)-(-)`. (Accessed 27-Sep-2023).

Yan, C., Tang, J., Jiang, B., and Fung, R. Y. (2015). Sequential appointment scheduling considering patient choice and service fairness. *International Journal of Production Research*, 53(24):7376–7395.

Yang, K. K., Lau, M. L., and Quek, S. A. (1998). A new appointment rule for a single-server, multiple-customer service system. *Naval Research Logistics*, 45(3):313–326.

Zacharias, C. and Armony, M. (2017). Joint panel sizing and appointment scheduling in outpatient care. *Management Science*, 63(11):3978–3997.

Zacharias, C. and Pinedo, M. (2017). Managing customer arrivals in service systems with multiple identical servers. *Manufacturing & Service Operations Management*, 19(4):639–656.

Zacharias, C. and Yunes, T. (2020). Multimodularity in the stochastic appointment scheduling problem with discrete arrival epochs. *Management science*, 66(2):744–763.

Zenteno, A. C., Carnes, T., Levi, R., Daily, B. J., and Dunn, P. F. (2016). Systematic OR block allocation at a large academic medical center. *Annals of Surgery*, 264(6):973–981.

Zhang, R., Han, X., Wang, R., Zhang, J., and Zhang, Y. (2022). Please don't make me wait! influence of customers' waiting preference and no-show behavior on appointment systems. *Production and Operations Management.*

Zhang, Y., Shen, S., and Erdogan, S. A. (2017). Distributionally robust appointment scheduling with moment-based ambiguity set. *Operations Research Letters*, 45(2):139–144.

Zhao, Y. (1994). Analysis of the GI^{X}/M/c model. *Queueing Systems*, 15(1):347–364.

Zhou, S. and Yue, Q. (2019). Appointment scheduling for multi-stage sequential service systems with stochastic service durations. *Computers & Operations Research*, 112:104757.

Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. (1997). Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4):550–560.

Zhu, S., Fan, W., Yang, S., Pei, J., and Pardalos, P. M. (2019). Operating room planning and surgical case scheduling: a review of literature. *Journal of Combinatorial Optimization*, 37:757–805.

Samenvatting

De wereldwijd stijgende kosten van de gezondheidszorg maken het noodzakelijk om middelen zo efficiënt mogelijk in te zetten. Eén manier om dit te doen is het optimaliseren van de vele roosters die binnen de gezondheidszorg gebruikt worden. Dit proefschrift richt zich specifiek op twee roosteringsproblemen binnen de gezondheidszorg: het *appointment scheduling problem* (afsprakenroosteringsprobleem) en het *master surgery scheduling problem* (chirugisch specialismenroosteringsprobleem). Het simpele *appointment scheduling problem* is een nuttig hulpmiddel, maar het heeft een beperkte rijkweidte: het houdt geen rekening met meerdere voorzieningen (zoals apparatuur, operatiekamers, of artsen) of met patiënten met verschillende kenmerken. Het *master surgery scheduling problem* is theoretisch goed bestudeerd, maar de toepassing ervan, en daarmee ook de verslaglegging over de toepassing ervan in de literatuur, is beperkt, zeker buiten de academische ziekenhuizen.

Dit proefschrift draagt bij aan de literatuur over efficiëntie in de gezondheidszorg door het analyseren van generalisaties van het *appointment scheduling problem* en het verkennen van toepassingen van *master surgery scheduling* technieken in de praktijk.

Methoden en resultaten

In dit proefschrift beschouwen we eerst een afsprakenrooster waarbij de kansverdelingen van bedieningstijden verschillen tussen patiënten. Dit roept de vraag op van *sequencing*: de volgorde waarin patiënten met verschillende kenmerken geroosterd zouden moeten worden. We ontwikkelen hiervoor een heuristiek en passen deze toe op twee roosteringsparadigma's. Deze heuristiek maakt het mogelijk om snel oplossingen te vinden en onderbouwt bepaalde volgorderegels. De effectiviteit van deze volgorderegels wordt bepaald, waarbij de conclusie wordt bevestigd dat de volgorde waarin patiënten aankomen minstens zoveel invloed heeft op de prestaties van een rooster als het bepalen van hun aankomsttijden.

Vervolgens laten we de aanname van *continuïteit van zorg* los, door toe te staan dat patiënten geholpen worden door verschillende zorgverleners, zogenaamde *pooling*. Dit maakt de *Lindley-recursie* irrelevant, en dus wordt gekozen voor fase-typeverdelingen om het systeem te modelleren. Deze verdelingen lijden echter aan een "vloek der dimensionaliteit", waarbij het aantal fasen dat nodig is om het systeem te beschrijven snel toeneemt en omslachtig wordt naarmate het aantal patiënten en zorgverleners toeneemt. Om deze reden wordt een vereenvoudigingstechniek toegepast die de omvang van het probleem verkleint, terwijl de informatie die nodig is voor optimalisatie behouden blijft. Een scenario met meerdere zorgverleners is moeilijk te analyseren, en we kunnen slechts tot het vermoeden komen dat er een uniek optimum bestaat wanneer de kansverdelingen van de bedieningstijden van patiënten unimodaal zijn. (Het is de moeite waard om te vermelden dat er een tegenvoorbeeld bestaat voor een bimodale verdeling). Opvallende resultaten worden gevonden zowel met betrekking tot de vorm van de optimale oplossingen, die afwijken van de typische koepelvorm, als de indrukwekkende besparingen die behaald kunnen worden door het *poolen* van zorgverleners. Het model wordt uitgebreid naar een *heavy traffic* setting, waarin patiënten dicht op elkaar arriveren. Dit levert analytische oplossingen op die eenvoudig te berekenen zijn en tegelijkertijd robuust tegen misspecificatie van de kansverdelingen van bedieningstijden. Het concept van *pooling* wordt verder uitgebreid door patiënten te laten arriveren in groepen van twee of meer - zogenaamde *batch arrivals.* Dit is bedoeld om tegemoet te komen aan een nadeel van het vorige scenario, waarbij patiënten één voor één aankomen en individuele artsen geen eigen afsprakenrooster hebben, hetgeen de planning van en controle over hun werkdag bemoeilijkt. Als startpunt analyseren we dit scenario in de *steady state*, waarbij we uitgaan van *exponentieel verdeelde* bedieningstijden. We laten zien dat de doelstellingsfunctie van dit voorbeeld van een scenario met *pooling* convex is, hetgeen ons een stap dichter bij een antwoord op de vraag van convexiteit brengt. We vergelijken de prestaties van roosters mét *batch arrivals* met die van roosters zonder *batch arrivals* en constateren dat de verwachte wacht- en *idle*-tijden verslechteren, maar slechts in beperkte mate. Ook laten we zien hoe dit probleem bestudeerd kan worden in een *transient setting*, waarbij we een methode gebruiken die later uitgebreid kan worden naar fasetypeverdelingen buiten de exponentiële verdeling.

Naast deze theoretische bijdrage aan de problematiek omtrent het roosteren van afspraken, kijken we ook naar de optimalisatie van *master surgery schedules* in de praktijk. We doen verslag van de ontwikkeling en implementatie van een *master surgery schedule* met behulp van lineaire optimalisatie. We richten ons hierbij in het bijzonder op het proces van het verzamelen en implementeren van beperkingen, en het managen van de verwachtingen van stakeholders. Het rooster werd geoptimaliseerd gedurende meerdere ontwerprondes, waarbij bij elke stap feedback vanuit het ziekenhuis werd meegenomen. Het project was een succes: het aantal *split blocks* (halve blokken), dagen waarop een chirurgisch specialisme slechts een halve in plaats van een hele dag kan opereren, werd gereduceerd van 40 naar 14 per cyclus van vier weken, en er werd voldaan aan een groot aantal beperkingen. Dit alles leidde tot een verhoging van de productiviteit in het gehele ziekenhuis. Om de implementatie van soortgelijke projecten elders te vergemakkelijken, delen we de *lessons learned* van dit project, waarbij we ingaan op zowel de succesfactoren als de valkuilen die we zijn tegengekomen.

Aanbevelingen

In dit proefschrift wordt gekeken naar het roosteren van patiëntafspraken in zowel de huisartsen- als de poliklinische zorg - *appointment scheduling* - en naar het roosteren van chirurgische specialismen binnen een ziekenhuis - *master surgery scheduling.* Op basis van dit onderzoek kunnen enkele aanbevelingen worden gedaan. In de eerste plaats stellen we voor om, wanneer de (relevante) gegevens beschikbaar zijn, de volgorde waarin patiënten geholpen worden te optimaliseren, zelfs als er alleen heuristische methoden beschikbaar zijn. In de tweede plaats adviseren we om waar mogelijk voorzieningen te bundelen (*poolen*); in dit geval zijn fase-typeverdelingen een uitstekend hulpmiddel om roosters te optimaliseren. Tot slot merken we op dat gevestigde methoden, zoals lineaire optimalisatie, nog onderbenut zijn in de gezondheidszorg, en dat deze het potentieel hebben om een groot verschil te maken. We moedigen zorginstellingen aan om na te denken over wat er in hun organisatie geoptimaliseerd zou kunnen worden en operationele researchers om de mogelijkheden voor optimalisatie onder de aandacht te brengen.

Acknowledgments

Most of all, I would like to thank Marleen for her support not only during the PhD, but during every day of the last fifteen years. She has been a tremendous source of strength through some very difficult times, and I owe all my achievements, academic and otherwise, to her.

I, of course, also owe a great deal to my family. I wish to thank my mother Rosemarie, who with luck may now refer to her son the doctor, my brother Martin, who is not a doctor, my father Gary, who instilled in me the importance of education, and Ginger, Seve, Izzy and Lobbus, who offered support in their own ways. To my family-in-law, in particular Marieke and Arie, I owe a debt of gratitude for taking me into their home and teaching me how to navigate life in the Netherlands.

Friends, both those on the outside as well as fellow inmates, also deserve thanks. There are too many to name in detail, so I will restrict the list to my longest serving colleagues. This of course includes my desk-mates and paranymphs: Ujjwal and Paulina, who provided endless entertainment; Rob, Leo and Yannik, my desk-mates at the start, who gladly took me into the fold, and who provided inspiration for sticking it out; and Martha, Felipe and Kayleigh, with whom many enjoyable conversations were had, grievances were discussed, and mutual support was shared. I must also thank Tijmen for his work proofreading this book.

Naturally, no such academic achievement would be possible without the guidance of (co-)promoters, nor the stimulating and supportive environment provided for by the School and all my colleagues, to whom I am grateful. I would especially like to thank Prof. Dr. ir. Dick den Hertog and Prof. Dr. Ronald Does for their guidance throughout the PhD, and Prof. Dr. Marc Salomon for his stewardship. I would like to give particular thanks to Dr. Alex Kuiper, with whom interesting discussions on all manner of topics were held before, and accumulated upon, a whiteboard, and who was always happy to provide genuine support.

Finally, I am grateful to all the members of the committee for taking the time to read my dissertation and to attend my defence.