Adaptive appointment scheduling with periodic updates

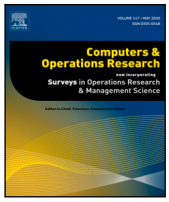
Mahes, R.; Mandjes, M.R.H.; Boon, M.

Computers & Operations Research 161 (2024) 106437

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor



Adaptive appointment scheduling with periodic updates

Roshan Mahes [a,b,*,1], Michel Mandjes [a,b,c,d,1], Marko Boon [c,e]

[a] Korteweg-de Vries Institute for Mathematics, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, the Netherlands
[b] Amsterdam Business School, Faculty of Economics and Business, University of Amsterdam, Amsterdam, the Netherlands
[c] Eurandom, Eindhoven University of Technology, Den Dolech 2, 5612 AZ Eindhoven, the Netherlands
[d] Mathematical Institute, Leiden University, P.O. Box 9512, 2300 RA Leiden, the Netherlands
[e] Department of Mathematics and Computer Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands

ARTICLE INFO

Dataset link: https://adaptiveschedule.eu.pythonanywhere.com/download/service_times.csv

Keywords:
Service systems
Appointment scheduling
Phase-type distribution

ABSTRACT

The classical paradigm in appointment scheduling is to rely on 'a priori schedules', determined by minimizing the given cost function; the corresponding arrival times are then announced to the clients, and not adjusted while serving them. The idea of the present paper is to reduce the cost by periodically updating the schedule (and notifying the clients about this), based on the current state of the system. Evaluation of the objective function is done highly efficiently and accurately by approximating the service times by their phase-type counterparts. The resulting method is computationally inexpensive, thus facilitating frequent evaluation and periodic adaptation of schedules 'on the fly'. A computational study illustrates the performance of the method, including an assessment of the impact of the rescheduling frequency and the variability of the service times. The most prominent conclusion is that typically, even with relatively few updates, costs can be reduced drastically. Our experiments, however, also reveal that one can construct instances for which increasing the rescheduling frequency does not guarantee a cost reduction; we provide an in-depth analysis of the remarkable phenomenon. The work has broad application potential, e.g., in healthcare and for delivery companies.

1. Introduction

In many service systems appointment scheduling plays a pivotal role. A schedule is (in its most basic form) an increasing sequence of arrival times $t_1, \ldots, t_n$ at which the n clients are supposed to arrive at the service facility. These times should be chosen such that the interests of the service provider and the clients are properly balanced. It is customary to measure the service provider's cost in terms of her idle time, and the clients' (aggregate) cost in terms of the sum of their waiting times. Our goal is to find a schedule that optimally balances the interests of both the service provider and her clients. On the one hand, the service provider wishes to efficiently run the system, in the sense that there is always a client in service. On the other hand, it is desired that the clients are provided a sufficiently high level of service, i.e., the waiting times should be as low as possible. Minimizing a cost function that encompasses both these idle times and waiting times over the arrival times provides us with an optimal schedule.

Classically, one works with what could be called 'a priori schedules': by minimizing the given objective function the arrival times $t_1, \ldots, t_n$ are determined, these are announced to the clients, and not adjusted while delivering the service. It is clear that with the advent of technologies that facilitate sending out notifications in real time, one can potentially do much better. When the service provider is behind schedule, she may want to postpone later appointments, whereas when she is ahead of schedule there is an incentive to bring them forward, so as to avoid excessive idle and waiting times. The idea is that one could, for instance periodically, adapt the schedule to reduce idle times and waiting times. The main goal of our paper is to develop a framework with which one can assess the potential gains of this type of adaptive scheduling, expressed in terms of a reduction of the cost function.

Typical examples of settings where appointment schedules are used can be found in the healthcare and parcel delivery context. In the healthcare setting, there is a service location (i.e., a clinic or medical practice) that is visited by patients. In the delivery context, idle time corresponds to the scenario that the deliverer arrives at the client's location before the announced time, while waiting time is the time a client has to wait after this scheduled time. The model dynamics featured in our work should be seen as a simplification of reality; our study mainly serves the goal of assessing the potential gain that can be achieved by updating schedules. This means that various domain-specific aspects are intentionally left out, such as (in the healthcare

* Corresponding author at: Korteweg-de Vries Institute for Mathematics, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, the Netherlands.
E-mail address: a.v.mahes@uva.nl (R. Mahes).

[1] Their research is partly funded by the NWO Gravitation project NETWORKS, grant number 024.002.003.

https://doi.org/10.1016/j.cor.2023.106437
Received 18 March 2023; Received in revised form 22 September 2023; Accepted 22 September 2023
Available online 25 September 2023

context) no-shows, walk-ins, availability of certain equipment, etc., and (in the delivery context) the fact that in practice one typically works with delivery windows rather than with delivery times.

In the setup we consider in this paper, the cost function we focus on is the sum of the mean idle times and the mean waiting times, but in principle, one could work with any cost function that is based on distributional properties of the idle times and waiting times (second moments, certain quantiles, etc.). Even though our approach allows us to reschedule at any time we want, for reasons of transparency we mainly consider periodic updates (i.e., the update epochs are equidistant in time). We in addition analyze alternative mechanisms, such as rescheduling after each client's start of service and rescheduling after each client arrival. Other practically relevant extensions that we consider are ones in which we impose restrictions on the set of admissible adapted schedules. This includes a study of the setting in which one cannot adjust the appointments that lie in a given time interval immediately after the update (e.g., within the first 30 min. after the rescheduling epoch).

We proceed by discussing the problem at a more technical level. At any rescheduling epoch, the state information that is available is (i) the number of clients who are currently waiting, (ii) the elapsed service time of the client in service (if any), and (iii) the number of clients who have not entered the system yet. We consider the (natural) setting in which we are not given the full service-time distributions, but just their means and variances. Following the approach advocated in e.g., Kuiper et al. (2015, 2022) and Tijms (1994), we identify phase-type distributions with this mean and variance. Phase-type distributions have various attractive properties, most notably they can be used to fit any distribution on $[0, \infty)$ arbitrarily closely (Asmussen, 2003, Theorem III.4.2) and they allow for relatively straightforward computations. They are defined as the absorption time of a suitably constructed continuous-time Markov chain with a given initial distribution $\boldsymbol{\gamma}$ and transition rate matrix T.

The framework developed thus combines two attractive features. In the first place, as pointed out above, whereas in the existing literature one predominantly focuses on static schedules, we here focus on adaptive schedules. We propose a procedure to update the appointment schedule, typically leading to a substantial reduction of the objective function. In the second place, our approach works with service-time distributions with any mean and variance, whereas in many appointment scheduling studies, exponentially distributed service times are assumed (Pegden and Rosenshine, 1990; Stein and Côté, 1994). This exponentiality assumption is often imposed to ease the analysis; indeed, the memoryless property facilitates symbolic computation of the objective function. The idea is that, by using a phase-type fit, still a relatively explicit evaluation is possible, i.e., in terms of a matrix-valued recursive scheme for the mean waiting and idle times, in the spirit of the one presented in Wang (1997).

Contributions. The paper has three main contributions. In the first place, we propose an adaptive scheduling framework, through which one can assess the gain due to rescheduling. More specifically, in the setup considered the schedule is periodically updated based on the state information available, viz. the number of clients in the system and the elapsed service time of the client in service (if any), besides the number of clients still to be scheduled.

In the second place, so as to determine the adapted schedule, a prerequisite is that we are able to compute the objective function given the state information. An important property that we rely upon is that the residual service time (given that the elapsed service time is, say, u) is again of phase type. The main implication of this fact is that the complexity of determining a schedule update is as high as the complexity of evaluating a static (i.e., a priori determined) schedule. We point out how the parameters of the phase-type distribution of the residual service time can be determined; concretely, it has the same transition rate matrix T as the service time itself, but an adjusted initial distribution $\boldsymbol{\gamma}$. We in addition show how the methodology developed in Wang (1997), which assumes homogeneous service times, can be generalized so as to also cover heterogeneous service times.

In the third place, we have performed a series of numerical experiments that primarily aim at quantifying the efficiency gain that can be achieved by adaptive scheduling (that is, relative to working with the static schedule). We provide a publicly available web tool by which adaptive schedules can be determined in real time. In our experiments we in particular study the impact of the rescheduling frequency. Generally, the more often the schedule is adapted the higher the gain, as expected; remarkably, even with relatively few updates, typically a substantial gain is achieved. One has to be careful though, in that one can construct instances in which a higher rescheduling frequency negatively impacts the gains; we provide an in-depth analysis of this counterintuitive phenomenon. We in addition investigate how the gain depends on (i) the variability of the service times and (ii) the cost function (in terms of the weight associated with the mean idle times relative to that of the mean waiting times). Then we consider several more realistic variants of our basic adaptive scheduling procedure. Most notably we assess the variant in which one cannot adjust the appointments that were scheduled directly after the rescheduling epoch. Also, an extra cost component is introduced to penalize deviations from the previously given schedule. Finally, we perform a comparison with the dynamic-programming-based rescheduling technique of Mahes et al. (2023), the most important conclusion being that our adaptive scheduling approach performs just slightly worse despite the fact that in the approach of Mahes et al. (2023) one optimizes over a larger decision space.

Literature. We proceed by providing a brief account of the literature in this area. We do not aim to provide an exhaustive overview; see e.g., Ahmadi-Javid et al. (2017) for a comprehensive, and relatively recent, survey. In Ahmadi-Javid et al. (2017), the authors distinguish between three levels: the strategic level (concerning design decisions, such as the choice of the number of servers), the tactical level (concerning planning decisions, such as the allocation of capacity to patient groups), and the operational level (e.g., focusing on determining the precise schedules). The theme of the current paper, the adaptation of schedules, clearly belongs to the operational measures but is hardly covered by Ahmadi-Javid et al. (2017).

The static scheduling problem, in which a schedule is determined a priori and not adapted, has been studied intensively. A methodology to find the optimal static schedule under specific distributional assumptions was presented in Wang (1997), while an extension to multiple servers can be found in Kuiper and Lee (2022). Various realistic features have been incorporated into the appointment scheduling problem. For example, in Chen and Robinson (2014) and Erdoğan and Denton (2013) one studies a combination of both routine clients that are assigned an appointment time in advance, and last-minute clients, seeking an appointment on the same day. A highly general framework, modeled as a nonlinear integer program, simultaneously covering no-shows, non-punctuality, and walk-ins, can be found in Zacharias and Yunes (2020). Another substantial part of the literature focuses on the client sequencing problem, i.e., optimizing the order of arrivals of the clients (which is assumed to be fixed in the framework studied in our paper) (Berg et al., 2014; Mak et al., 2015). Concretely, ordering the clients in increasing variance of their service-time distributions appears to be the most common rule to produce a good sequence (de Kemp et al., 2021; Kong et al., 2016). The exact optimal sequencing policy is still unknown when there are three or more clients (Mak et al., 2014).

The present paper is related to the dynamic programming approach developed in Mahes et al. (2023). There a setting is studied where at each client arrival the arrival time of the next client is determined. It is shown that this results in a significant reduction in cost as compared to static scheduling. In the current paper, we comment on the performance of the adaptive scheduling approach relative to the one of Mahes et al. (2023).

Unlike in our work, the term 'adaptive appointment scheduling' is in the literature frequently referring to updating schedules by sequentially finding an available time slot adapted to the clients' needs. For example, Wang and Fung (2014, 2015) and Wang and Gupta (2011) dynamically learn clients' preferences as appointment requests come in, while Erdoğan et al. (2015) uses stochastic integer programming and Doğru and Melouk (2019) proposes a simulation optimization approach. In all these papers, existing appointments are not modified when a schedule gets updated, rather available slots are filled according to some policy.

Even though most work on appointment scheduling focuses on healthcare applications, it can be applied in a substantially broader context. Other settings in which it can be applied include that of clients and a consulting professional, automobiles and a service center, and legal cases and a courtroom (Robinson and Chen, 2003). Over the past few years, due to the increasing presence of delivery services of various sorts, there has been a strong focus on appointment scheduling in a spatial setting, i.e., involving a routing component. These problems are intrinsically hard, as they combine all complications arising in appointment scheduling with those of the traveling salesman problem, see e.g., Liu et al. (2019) and Zhan et al. (2021). Moreover, during the delivery process, the driver might choose to take a different path from the prescribed route, a challenge addressed in Ghosh et al. (2023). In addition, Decerle et al. (2018) considers a routing and scheduling problem with time window constraints, based on the clients' availability. A way to assign such time windows before creating a vehicle routing schedule is described in e.g., Spliet and Desaulniers (2015) and Spliet and Gabor (2014).

Organization. The structure of this paper is as follows. In Section 2 we formally define the cost function and the objective. The evaluation of this cost function is described in Section 3. The performance of the method is assessed in Section 4. The paper is concluded with some final remarks.

2. Adaptive scheduling procedure

In this section, we formally describe the problem that is considered in this paper, as well as the algorithm that we propose. The structure of the section is as follows. In Section 2.1 we describe an auxiliary scheduling problem that will form the main building block of our algorithm. A crucial step is that we express our cost function in terms of the clients' mean sojourn times. Then, in Section 2.2, we describe our algorithm, in which we adapt the schedule at equidistant points. It is worth noting that in Section 3 we will point out how the objective function, featured in the algorithm, can be evaluated.

2.1. An auxiliary scheduling problem

We start by explaining a specific static scheduling problem that will, in Section 2.2, serve as the basis of our adaptive scheduling algorithm. To this end, we consider a sequence of $n \in \mathbb{N}$ clients with service times represented by the non-negative, independent random variables $B_1, \dots, B_n$; note that these are not necessarily identically distributed. Our goal is to find a schedule $t_1, \dots, t_n$, where t_i represents the arrival time of the i-th client (i.e., we implicitly require $0 \leqslant t_1 \leqslant \cdots \leqslant t_n$).

In this paper, we optimize an objective function that balances the interests of both the clients and the service provider. Concretely, we consider the following optimization problem:

$$\min_{t_1,\dots,t_n} \; f(t_1, \dots, t_n \mid k, u), \tag{1}$$

with, for $\omega \in (0,1)$,

$$f(t_1, \dots, t_n \mid k, u) := \omega \sum_{i=1}^{n} \mathbb{E}I_i + (1-\omega) \sum_{i=1}^{n} \mathbb{E}W_i,$$

where I_i and W_i are the idle and waiting time associated with client i, respectively; we refer to Fig. 1 for an illustration. Note that the entries

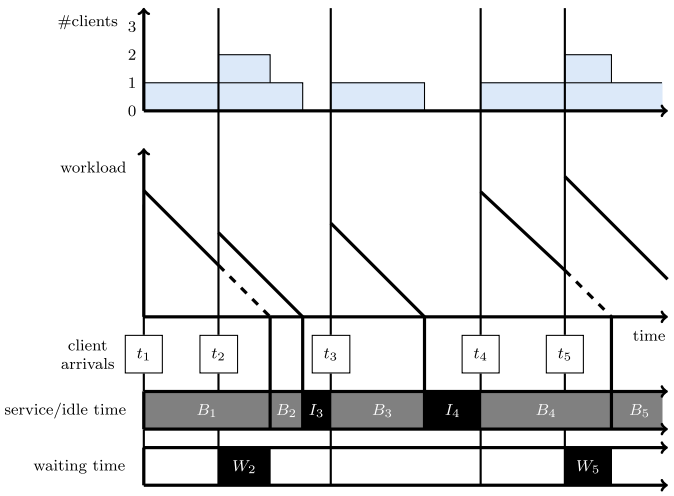


Fig. 1. Key quantities in appointment scheduling. Figure courtesy of Kuiper (2016).

of the sequences $(\mathbb{E}I_i)_{i=1}^n$ and $(\mathbb{E}W_i)_{i=1}^n$ are implicit functions of the arrival times $t_1, \dots, t_n$. In the conventional version of this problem, the goal is to select $t_1, \dots, t_n$ that minimize the above cost function. The setting of our auxiliary scheduling problem, however, is different in the sense that at every rescheduling epoch, we are given the following *state information*:

- The number of clients $k \in \{0, 1, \dots, n\}$ who have already entered the system at time 0. This thus means that, if $k > 0$, then $t_1 = \cdots = t_k = 0$. It is clear that also if $k = 0$, then we should take $t_1 = 0$: bearing in mind the objective function that we wish to minimize, it is pointless to let client 1 enter after time 0.
- The value of the elapsed service time u of client 1 if $k > 0$ (if $k = 0$, then we set $u \equiv 0$). This means that the remaining service time of the client in service is distributed as $B_1 - u$ conditional on $B_1 > u$. As we hold u fixed (in this subsection), we will sometimes denote this remaining service time simply by B_1 (or by $B_1(u)$, to stress the dependence on u).
- In total, there are n clients that remain to be served, of which the first k already entered the system. This means that there are $n-k$ clients to be scheduled.

It turns out to be convenient to somewhat rewrite the objective function. Let S_i the *sojourn time* of client i, i.e., her waiting time W_i increased by her service time B_i. Note that the time at which client i leaves the system is equal to the sum of all service and idle times corresponding to all clients up to and including this client. Hence, for $j = 1, \dots, n$,

$$\sum_{i=1}^{j} B_i + \sum_{i=1}^{j} I_i = t_j + S_j;$$

a pictorial illustration of this fact is provided by Fig. 1. The above entails that we can express the (expected) waiting and idle times in terms of the (expected) sojourn times: naturally, $\mathbb{E}W_1 = \mathbb{E}I_1 = 0$, and, for $i = 2, \dots, n$,

$$\mathbb{E}W_i = \mathbb{E}S_i - \mathbb{E}B_i, \quad \mathbb{E}I_i = t_i + \mathbb{E}W_i - (t_{i-1} + \mathbb{E}S_{i-1}),$$

the latter of which can be interpreted as the expected time difference between the service completion of client $i-1$ and the start of service of client i. As the expected service times $\mathbb{E}B_i$ are given numbers, we now have rewritten our problem in terms of the expected sojourn times of the clients only, a fact that we will extensively exploit in this paper. In the sequel, we (informally) denote the above optimization procedure by

$$\texttt{Schedule}(B_1, B_2, \dots, B_n \mid k, u) \mapsto (t_1, \dots, t_n),$$

where, as mentioned, the distribution of (the remaining part of) B_1 depends on the elapsed service time u of client 1. In Section 3 we point out how the routine `Schedule` can be evaluated; in Section 2.2 we explain how we can use `Schedule` in the key problem studied in this paper, namely that of periodically adapting the appointment schedule.

2.2. Periodically updating the schedule

The main idea is to update the schedule every Δ time units, for some predefined interval length $\Delta > 0$. At the m-th update, at time $\tau_m := m\Delta$, the state information available is the number of clients k_m in the system at time τ_m, as well as (if $k_m > 0$) the elapsed service time u_m of the client in service. The underlying thought is that it allows us to respond to situations in which we are ahead of or behind schedule. One would expect that the smaller the value of Δ, the more frequently we adjust the schedule, and the lower the cost; we explore this relationship in great detail in the numerical experiments of Section 4.

Below we include pseudocode for our periodically updated schedule. The main idea is to rerun the routine `Schedule`, as was introduced above, at the times τ_m, with the current state information as input. It uses the following functions:

- `NumberInSystem`(t) provides the number of clients in the system at time $t \geqslant 0$, i.e., covering waiting clients (if any) as well as the client in service (if any).
- `NumberServed`(t) provides the number of service completions in $[0, t]$ for $t \geqslant 0$.
- `Elapsed`(t) provides the elapsed service time of the client in service at time $t \geqslant 0$ (and 0 if there is no client in service).

Algorithm 1 describes, in self-evident pseudocode, how we update the schedule at the time epochs τ_m. In the algorithm, the variable k keeps track of the current number of clients in the system, u denotes the elapsed service time of the current client in service (if any, and otherwise $u := 0$), and ℓ is the number of clients served so far. In addition, $\mathbf{1}_i$ is the all-ones vector of dimension $i \in \mathbb{N}$.

Algorithm 1: `PeriodicAdaptiveScheduling`$(B_1, \ldots, B_n, \Delta)$

Result: Periodically adapted schedule, with updates at times $\tau_m = m\Delta$

Initialization: $k_0 := 0$; $u_0 := 0$; $\ell_0 := 0$; $m := 0$; $t_n := \infty$
while $t_n > \tau_m$ **do**
  $(t_{\ell_m+1}, \ldots, t_n) :=$ `Schedule`$(B_{\ell_m+1}, \ldots, B_n \mid k_m, u_m) + \tau_m \cdot \mathbf{1}_{n-\ell_m}$
  $m := m + 1$
  $k_m :=$ `NumberInSystem`(τ_m)
  $\ell_m :=$ `NumberServed`(τ_m)
  $u_m :=$ `Elapsed`(τ_m)
end

In the above description, for ease we let the τ_m correspond to equidistant points in time, but it is clear that in principle non-equidistant updates are possible, too. In particular, it is conceivable that early in the schedule, the amount of uncertainty is still modest, so there is a good reason to take τ_1 relatively large. We do not explore such non-equidistant updates in this paper.

3. Evaluation of the objective function

Now that we have described our adaptive scheduling approach, we continue by pointing out how our objective function $f(t_1, \ldots, t_n \mid k, u)$ can be evaluated. In our setup, we suppose that we know the first two moments of the service times, or, equivalently,

$$\big(\mathbb{E}B_1, \ldots, \mathbb{E}B_n\big), \quad \big(\mathbb{V}\mathrm{ar}(B_1), \ldots, \mathbb{V}\mathrm{ar}(B_n)\big).$$

We do so following a well-established approach: fitting so-called *phase-type* distributed random variables (Kuiper et al., 2015, 2022), and then applying a technique in the spirit of the one developed in Wang (1997) to compute the expected sojourn times $\mathbb{E}S_i$. A complication is that the procedure relied on in Kuiper et al. (2015, 2022) must be adapted to incorporate the k clients present at time 0, and if $k > 0$ the elapsed service time u of the client in service. In Section 3.1 we briefly recall the way to map a pair $(\mathbb{E}B_i, \mathbb{V}\mathrm{ar}(B_i))$ on a phase-type distribution; as we will see two special classes of phase-type distributions play a key role here. Section 3.2 describes how to evaluate the cost function $f(t_1, \ldots, t_n \mid k, u)$ for a given schedule $(t_1, \ldots, t_n)$ (where, evidently, the components of this vector are non-decreasing), thus solving the above-mentioned complication. In Section 3.3 we comment on minimizing the cost function over $(t_1, \ldots, t_n)$.

3.1. Phase-type fit

If the service times would have been exponentially distributed, then it would be relatively straightforward to devise a procedure to compute the expected sojourn times $\mathbb{E}S_i$, essentially owing to the fact that the number of clients in the system is a continuous-time Markov chain. For an exponentially distributed service time B_i, the *squared coefficient of variation* (in the sequel abbreviated to SCV)

$$\mathbb{S}(B_i) := \frac{\mathbb{V}\mathrm{ar}(B_i)}{(\mathbb{E}B_i)^2}$$

equals 1, entailing that necessarily the mean and standard deviation match. In various application domains, however, data analysis has revealed that service times substantially deviate from being exponential. In particular, in medical applications (Çayırlı et al., 2006) often the service times are relatively deterministic, reflected in the corresponding SCV being smaller than 1. Clearly, assuming exponential service times would provide suboptimal scheduling rules.

The above complication is remedied by working with specific convenient phase-type distributions by which we cover all values of the SCV. Phase-type distributions can be seen as generalizations of the exponential distribution that still allow a fairly explicit analysis. Formally, a phase-type distribution can be characterized as follows. Consider a continuous-time Markov chain $\{X_t\}_{t\geqslant 0}$ with state space $E = \{1, \ldots, d+1\}$, where states $1, \ldots, d$ are transient and state $d+1$ is absorbing. The initial state $X_0 \in \{1, \ldots, d\}$ is sampled according to a probability (row) vector $\boldsymbol{\gamma} \in \mathbb{R}^d$, i.e., its entries are non-negative and sum to 1. The process has a transition rate matrix of the form

$$Q = \begin{bmatrix} T & \boldsymbol{t} \\ \mathbf{0}_{1\times d} & 0 \end{bmatrix},$$

with $T \in \mathbb{R}^{d\times d}$, $\mathbf{0}_{i\times j}$ denoting an all-zeroes matrix of dimension $i\times j$, exit rate vector $\boldsymbol{t} := -T\mathbf{1}_d$, and $\mathbf{1}_d$ a d-dimensional all-ones (column) vector. The time it takes to reach the absorbing state, i.e.,

$$B := \inf\{t > 0 \mid X_t = d+1\},$$

is called a phase-type distributed random variable with initial distribution $\boldsymbol{\gamma}$ and subintensity matrix T, denoted by $B \sim \mathrm{PH}_d(\boldsymbol{\gamma}, T)$. The exponentially distributed times spent in each of the states $1, \ldots, d$ are typically referred to as *phases*, explaining the terminology 'phase-type'.

The phase-type distribution owes its popularity to two features. In the first place, as already mentioned above, they are attractive from a computational point of view: relying on standard tools from linear algebra, the corresponding densities and distribution functions can be numerically evaluated. In the second place, any distribution on the positive halfline can be approximated arbitrarily closely by a phase-type distribution (Asmussen, 2003, Theorem III.4.2). As extensively motivated in Tijms (1994), and used in e.g., Kuiper et al. (2015) and Mahes et al. (2023), two specific subclasses are of particular interest: mixtures of Erlang distributions and hyperexponential distributions. More concretely, with these two subclasses, all values of the SCV are covered, while at the same time, they have the attractive feature that they are low-dimensional (in terms of the number of parameters). In particular, in the context of appointment scheduling, numerical

evaluation revealed that the error introduced by replacing the true service-type distribution by these specific subclasses has turned out to be negligible; cf. the findings reported in Kuiper (2016, pp. 110–111) and Mahes et al. (2023).

We continue by briefly discussing mixtures of Erlang distributions and hyperexponential distributions, and in particular, we describe how to map the mean and SCV of a random variable B on the corresponding parameters. A more extensive account of this two-moment fit is provided in e.g., Kuiper et al. (2015, 2022).

— *Case 1: SCV smaller than 1.* If the SCV is below 1, we approximate the non-negative random variable B by a mixture of Erlang distributions (or: a weighted Erlang distribution). To this end, denote by $\mathrm{E}(K,\mu)$ an Erlang distributed random variable with shape parameter K and scale parameter μ, and by U an independent uniform random variable on $[0,1]$. Then, for some $K\in\mathbb{N}$, $\mu>0$ and $p\in[0,1]$,

$$B\sim \mathrm{E}(K,\mu)\mathbb{1}_{\{U<p\}}+\mathrm{E}(K+1,\mu)\mathbb{1}_{\{U>p\}}.$$

In other words: with probability p the random variable B equals an Erlang-distributed random variable with K phases, and with probability $1-p$ an Erlang-distributed random variable with $K+1$ phases. The parameters are uniquely determined (Mahes et al., 2023; Tijms, 1994): we have

$$K=\left\lfloor\frac{1}{\mathbb{S}(B)}\right\rfloor,\quad p=\frac{(K+1)\,\mathbb{S}(B)-\sqrt{(K+1)(1-K\mathbb{S}(B))}}{\mathbb{S}(B)+1},\quad \mu=\frac{K+1-p}{\mathbb{E}B}.$$

Indeed, B is a phase-type distributed random variable. The corresponding subintensity matrix $T\in\mathbb{R}^{(K+1)\times(K+1)}$ is

$$T=\begin{bmatrix}-\mu & \mu & 0 & \cdots & 0\\ 0 & -\mu & \mu & \ddots & \vdots\\ \vdots & \ddots & \ddots & \ddots & 0\\ \vdots & & \ddots & -\mu & \mu(1-p)\\ 0 & \cdots & \cdots & 0 & -\mu\end{bmatrix}.$$

In addition, the initial distribution is given by $\boldsymbol{\gamma}=(1,0,\ldots,0)$.

— *Case 2: SCV larger than 1.* If the SCV is above 1, we approximate the non-negative random variable B by a hyperexponential distribution. For some $\mu_1,\mu_2>0$ and $p\in[0,1]$, in self-evident notation,

$$B\sim \mathrm{Exp}(\mu_1)\mathbb{1}_{\{U<p\}}+\mathrm{Exp}(\mu_2)\mathbb{1}_{\{U>p\}}.$$

Hence, B now equals with probability p an exponentially distributed random variable with mean μ_1^{-1}, and with probability $1-p$ an exponentially distributed random variable with mean μ_2^{-1}. To fit the mean and SCV, the parameters are not uniquely determined. This can be solved by imposing the *balanced means* condition (Kuiper et al., 2015; Tijms, 1994), i.e., $\mu_1=2p\mu$ and $\mu_2=2(1-p)\mu$ for some $\mu>0$. Using this, we find

$$p=\frac{1}{2}\left(1+\sqrt{\frac{\mathbb{S}(B)-1}{\mathbb{S}(B)+1}}\right),\quad \mu_1=\frac{2p}{\mathbb{E}B},\quad \mu_2=\frac{2(1-p)}{\mathbb{E}B}.$$

It is left to characterize B as a phase-type random variable. It is clear that the subintensity matrix $T\in\mathbb{R}^{2\times2}$ we have is

$$T=\begin{bmatrix}-\mu_1 & 0\\ 0 & -\mu_2\end{bmatrix},$$

and $\boldsymbol{\gamma}=(p,1-p)$.

We proceed by explaining how these phase-type distributions are used in our framework. The main idea is to replace each of the service times B_i by a phase-type distributed random variable, following the recipe described above, leading to a description of the form $B_i\sim\mathrm{PH}_{d_i}(\boldsymbol{\gamma}_i,T_i)$. In case the number of clients in the system at an observation epoch τ_m, previously denoted by k, is positive, however, we wish to take into account the elapsed service time u of the client in service. This can be done by the following procedure; we again distinguish between the service time being a mixture of Erlangs and hyperexponential.

It can be seen in both cases the distribution of B_i-u conditional on $B_i>u$, for some $u>0$, is still of phase type, with the same T_i as the one of B_i itself, but with a different initial distribution which now depends on the elapsed service time u (therefore denoted by $\boldsymbol{\gamma}_i(u)$). Let the process $\{X_{t,i}\}_{t\geqslant0}$, for $i=1,\ldots,n$, the (d_i+1)-dimensional continuous-time Markov chain corresponding to $B_i\sim\mathrm{PH}_{d_i}(\boldsymbol{\gamma}_i,T_i)$. Our objective is to find an expression for the j-th entry of $\boldsymbol{\gamma}_i(u)$, in the sequel denoted by $\gamma_{ij}(u)$. Note that $\gamma_{ij}(u)$ can be interpreted as $\mathbb{P}(X_{u,i}=j\mid B_i>u)$.

First consider the case that B_i is mixed Erlang, say, with parameters K_i, μ_i and p_i (so that $d_i=K_i+1$). As argued in Mahes et al. (2023), and as can be verified in a straightforward manner, the j-th entry of $\boldsymbol{\gamma}_i(u)$ equals $\gamma_{ij}(u)=\gamma_{ij}^{\circ}(u)/\gamma_i^{\circ}(u)$, where

$$\gamma_i^{\circ}(u):=\mathbb{P}(B_i>u)=\sum_{j=1}^{K_i}e^{-\mu_i u}\frac{(\mu_i u)^{j-1}}{(j-1)!}+(1-p_i)e^{-\mu_i u}\frac{(\mu_i u)^{K_i}}{K_i!},$$

$$\gamma_{ij}^{\circ}(u):=\mathbb{P}(X_{u,i}=j,B_i>u)=e^{-\mu_i u}\frac{(\mu_i u)^{j-1}}{(j-1)!}\mathbb{1}_{\{j=1,\ldots,K_i\}}+(1-p_i)e^{-\mu_i u}\frac{(\mu_i u)^{K_i}}{K_i!}\mathbb{1}_{\{j=K_i+1\}}.$$

Now consider the case that B_i is hyperexponential, say, with parameters μ_{i1}, μ_{i2} and p_i (so that $d_i=2$). Again writing $\gamma_{ij}(u):=\gamma_{ij}^{\circ}(u)/\gamma_i^{\circ}(u)$, it is readily checked that

$$\gamma_i^{\circ}(u):=\mathbb{P}(B_i>u)=p_ie^{-\mu_{i1}u}+(1-p_i)e^{-\mu_{i2}u},$$
$$\gamma_{ij}^{\circ}(u):=\mathbb{P}(X_{u,i}=j,B_i>u)=p_ie^{-\mu_{i1}u}\mathbb{1}_{\{j=1\}}+(1-p_i)e^{-\mu_{i2}u}\mathbb{1}_{\{j=2\}}.$$

Observe that when the elapsed service time equals 0, in both the mixed Erlang and hyperexponential case, we obtain that $\boldsymbol{\gamma}_i(0)=\boldsymbol{\gamma}_i$, as it should.

3.2. Computation of mean sojourn times

Our next task is to compute $f(t_1,\ldots,t_n\mid k,u)$ given that $B_i\sim\mathrm{PH}_{d_i}(\boldsymbol{\gamma}_i,T_i)$. As indicated earlier, if $k>0$, then $u\geqslant0$, and B_1 is to be understood as $B_1(u):=B_1-u\mid B_1>u$, which is of the type $\mathrm{PH}_{d_1}(\boldsymbol{\gamma}_1(u),T_1)$.

Recall that we managed to rewrite the objective function in terms of the expected values of the sojourn times S_i, $i=1,\ldots,n$. As will become clear below, we actually derive expressions for the full distribution function of the S_i, from which the means $\mathbb{E}S_i$ can be found in an evident manner. The procedure presented borrows elements from the one developed in Wang (1997). Importantly, Wang (1997) covers the case of i.i.d. service times only; below we point out how it extends to the case of heterogeneous (but still independent) clients.

Given the schedule $t_1,\ldots,t_n$, let $x_i:=t_i-t_{i-1}$ the i-th interarrival time, where we set $x_1:=t_1$. Denote by $N_i(t)$ the number of clients in the system at *shifted* time $t\in[0,x_{i+1})$, that is, t time units after the arrival of client i. Also, let $Z_i(t)$ be the phase the client in service is in at the same (shifted) time t. If the system is idle, i.e., if no client is in service, set $Z_i(t)=0$. Denote for $i=1,\ldots,n$, $k=1,\ldots,i$ and $z=1,\ldots,d_{i-k+1}$ the probability that the system is in state (k,z) at shifted time t by

$$p_{kz}^{(i)}(t):=\mathbb{P}(N_i(t)=k,Z_i(t)=z);$$

observe that if $N_i(t)=k$, then the index of the client in service is $i-k+1$.

Note that we define this probability for all states except state $(0,0)$, in which the system is idle, i.e., the only case in which at time t the i-th client has been served. Using this observation, the sojourn-time distribution $F_i(t)$ of the i-th client equals, with $\mathbf{1}_d$ denoting an all-ones vector of dimension $d\in\mathbb{N}$,

$$F_i(t):=\mathbb{P}(S_i\leqslant t)=1-\sum_{k=1}^{i}\sum_{z=1}^{d_{i-k+1}}p_{kz}^{(i)}(t)=1-\boldsymbol{P}_i(t)\mathbf{1}_{\sum_{k=1}^{i}d_k},$$

where we define the vector

$$\boldsymbol{P}_i(t):=\lim_{s\uparrow t}\left(p_{i,1}^{(i)}(s),\ldots,p_{i,d_1}^{(i)}(s),p_{i-1,1}^{(i)}(s),\ldots,\right.$$

$$p^{(i)}_{i-1,d_2}(s), \dots, p^{(i)}_{1,1}(s), \dots, p^{(i)}_{1,d_i}(s)\Big).$$

As the first client immediately gets served at time $t_1 = 0$, it directly follows that $\boldsymbol{P}_1(t) = \boldsymbol{\gamma}_1 \exp(\boldsymbol{V}_1 t)$ and $\mathbb{E}S_1 = -\boldsymbol{\gamma}_1 \boldsymbol{V}_1^{-1}\mathbf{1}_{d_1}$ for $t \in [0, x_2)$, with $\boldsymbol{V}_1 := T_1$; see e.g., Neuts (1994). When the second client arrives at time x_2, two scenarios are possible. Either the first client is still in service and the second client needs to wait, or the service of the first client has been completed (with probability $F_1(x_2)$) and the second client immediately goes into service. The service process can now be represented by the subintensity matrix

$$\boldsymbol{V}_2 := \begin{bmatrix} T_1 & (-T_1\mathbf{1}_{d_1})\boldsymbol{\gamma}_2 \\ \mathbf{0}_{d_2\times d_1} & T_2 \end{bmatrix};$$

when checking the compatibility of the vectors and matrices, realize that $\boldsymbol{\gamma}_i$ represents a d_i-dimensional *row* vector, entailing that $(-T_1\mathbf{1}_{d_1})\boldsymbol{\gamma}_2$ is of dimension $d_1 \times d_2$. We thus find for $t \in [0, x_3)$,

$$\boldsymbol{P}_2(t) = \left[\boldsymbol{P}_1(x_2), \boldsymbol{\gamma}_2 F_1(x_2)\right] \exp(\boldsymbol{V}_2 t),$$
$$\mathbb{E}S_2 = -\left[\boldsymbol{P}_1(x_2), \boldsymbol{\gamma}_2 F_1(x_2)\right] \boldsymbol{V}_2^{-1}\mathbf{1}_{d_1+d_2},$$

Continuing along these lines, we end up with the following recursive procedure.

Proposition 1. *Let $B_i \sim PH_{d_i}(\boldsymbol{\gamma}_i, T_i)$ be the service time and let x_i be the interarrival time corresponding to client $i = 1, \dots, n$. Set $D_i := \sum_{k=1}^{i} d_k$. Then, the sojourn-time distribution of client i is given by*

$$F_i(t) := \mathbb{P}(S_i \leqslant t) = 1 - \boldsymbol{P}_i(t)\mathbf{1}_{D_i},$$

where, for $t \geqslant 0$,

$$\boldsymbol{P}_i(t) := \boldsymbol{G}_i \exp(\boldsymbol{V}_i t).$$

Here, $\boldsymbol{G}_1 := \gamma_1$ and $\boldsymbol{V}_1 := T_1$, and, for $i = 2, \dots, n$,

$$\boldsymbol{G}_i := \left[\boldsymbol{P}_{i-1}(x_i), \boldsymbol{\gamma}_i F_{i-1}(x_i)\right],$$
$$\boldsymbol{V}_i := \left[\begin{array}{c|c} \boldsymbol{V}_{i-1} & \begin{matrix}\mathbf{0}_{D_{i-2}\times d_i} \\ (-T_{i-1}\mathbf{1}_{d_{i-1}})\boldsymbol{\gamma}_i\end{matrix} \\ \hline \mathbf{0}_{d_i\times D_{i-1}} & T_i \end{array}\right].$$

The expected sojourn time of client i equals

$$\mathbb{E}S_i = -\boldsymbol{G}_i \boldsymbol{V}_i^{-1}\mathbf{1}_{D_i}.$$

3.3. Optimizing the objective function

Now that we know how to evaluate the objective function, we briefly comment on optimizing it over the schedule $t_1, \dots, t_n$. A first remark is that this optimization problem can be expressed in terms of a convex programming problem, as has been rigorously established in Kuiper et al. (2022); this in particular entails that there is just one local minimum (which therefore is the global minimum as well). As a consequence, standard (quasi-)Newton minimization routines can be used to efficiently identify the minimum of $f(t_1, \dots, t_n \mid k, u)$ over time epochs $t_1, \dots, t_n$ such that $0 = t_1 = \cdots = t_k \leqslant t_{k+1} \leqslant \cdots \leqslant t_n$; here we set the arrival times of the first k clients to 0 as they already have entered the system. Essentially all standard numerical packages have implementations of state-of-the-art minimization routines that can quickly and accurately determine the corresponding minimizer; in our software, we have used the Sequential Least Squares Programming (SLSQP) solver implemented in the open-source Python package `SciPy`.

Remark 1. In our setup, appointment times can take continuous values, which is a leading paradigm in the appointment scheduling literature; see e.g., Kuiper et al. (2015), Kuiper and Lee (2022), Mak et al. (2015) and Wang (1997) and many of the approaches discussed in the overview (Ahmadi-Javid et al., 2017). In some practical single-server appointment systems, however, clients are assigned to time slots (i.e., intervals whose lengths are multiples of some given granularity, for instance, five minutes). An elementary way to convert continuous-time schedules into slotted counterparts is by rounding off, but one could resort to more sophisticated procedures as well (e.g., by searching a discrete set of gridpoints around the optimal continuous-time schedule).

4. Numerical evaluation

In this section, we assess the performance of our approach through a series of numerical experiments. We start by describing, in Section 4.1, the interface of the applet we developed. Assessment of the standard variant is covered by Section 4.2, more practical variants are considered in Section 4.3, and a comparison with alternative rescheduling methods is given in Section 4.4.

4.1. Applet

We have developed a user-friendly applet that optimizes our objective function using the open-source solver SLSQP mentioned in Section 3.3. By this noncommercial applet, any user can adopt our methodology without having to do any coding. It enables users to compute the optimal schedule, essentially in real time, for any instance the optimal schedule. Here, an instance is a combination of the number of clients to be served n, the mean $\mathbb{E}B_i$ and SCV $\mathbb{S}(B_i)$ of the service time of each client $i = 1, \dots, n$, the weight $\omega \in (0, 1)$, the state information, viz. the number of clients in the system $k \in \{0, \dots, n-1\}$ and the elapsed service time $u \geq 0$ of the client in service (if any). Note that this applet allows a user to update her schedule not only periodically, but also at any other desired time.

In practical situations, one may not want to adapt the appointments that lie in the interval immediately after the rescheduling epoch (for instance in situations in which clients may already be on their way to the service location). To deal with this issue, we provide the option to leave the schedule of all clients fixed until a certain time $\tau \geq 0$ (see Experiment 4). In this case, the user also enters the appointment times of the clients to show up before time τ as generated by the existing schedule, where the (current) time in which we observe the state information is considered to be time 0. The interface of the applet is displayed in Fig. 2. It can be accessed through https://adaptiveschedule.eu.pythonanywhere.com.

4.2. Standard variant

In this subsection, we consider the standard variant of our model, with the objective of quantifying the efficiency gain of our adaptive method relative to static scheduling. It is important to note that the value of the cost function for the adaptive approach cannot be directly computed, as opposed to the cost of static scheduling. More precisely,

- the cost of the static schedule can be directly obtained by numerically solving the optimization problem formulated in Eq. (1), applying the techniques developed in e.g., Kuiper et al. (2015) and Wang (1997), which is basically just executing Algorithm 1 with only one iteration of the while-loop;
- the adaptive schedule, however, is recomputed at the time epochs $\tau_m = m\Delta$, and therefore depends on the precise state of the system at these epochs. With our procedure, at every time epoch τ_m, only the cost of the remaining schedule is evaluated as if the schedule will not be updated again.

To evaluate the cost of the adaptive schedule, we estimate the cost of the adaptive approach relying on Monte Carlo simulation. We note however, as discussed in greater detail in Appendix A, that in a few simple cases (assuming exponentially distributed service times and a low number of clients) a somewhat more explicit approach can be followed.

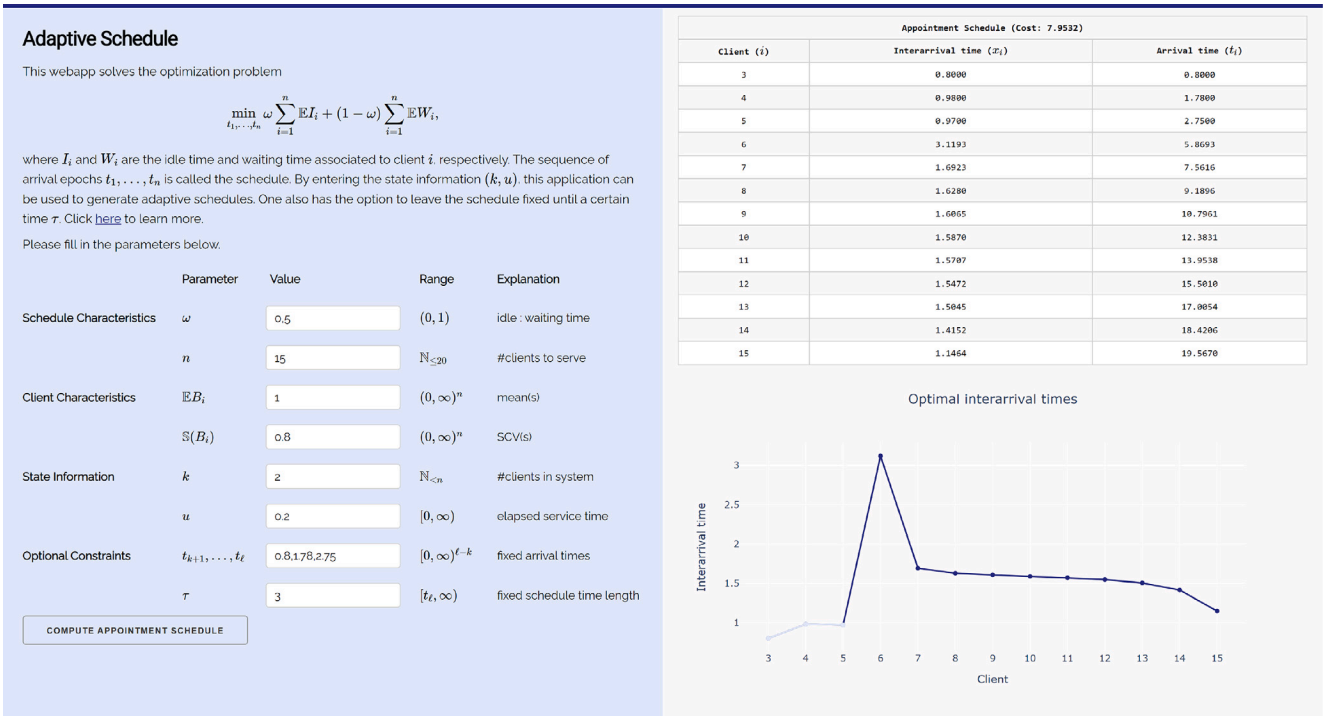


Fig. 2. Interface of applet.

We proceed by providing a more detailed discussion of the above-mentioned simulation-based approach to evaluate the cost of the adaptive approach. In every simulation run the service times B_1 up to B_n are sampled given their (phase-type) distributions, and at the epochs τ_m the schedule is reevaluated, depending on the state information available at these moments. In the end, given the final schedule, the realized idle and waiting times, and therefore the value of the cost function for this run can be computed using the Lindley recursion (Lindley, 1952). The pseudocode for a simulation run can be found in Appendix B. Performing a sizeable number of runs, and averaging the realized costs, we can accurately estimate the cost corresponding to our adaptive approach. In this section, we denote the cost of the static method by C_{stat}, the cost of the adaptive method (with rescheduling time Δ) by $C_{\text{adapt}}(\Delta)$ and the efficiency gain by

$$\Gamma(\Delta) := \frac{C_{\text{stat}} - C_{\text{adapt}}(\Delta)}{C_{\text{stat}}}. \tag{2}$$

For reasons of transparency, the experiments are organized such that we vary the parameters as much as possible in an isolated manner. This way, we obtain insight into the impact of each of these parameters on the efficiency gain. Unless otherwise stated, experiments are based on $N = 10^6$ simulation runs and are performed on the National Supercomputer Snellius supported by SURF (www.surf.nl).

Experiment 1 (*Effect of the Parameters*). We start by evaluating how the rescheduling period Δ affects the cost of the schedule. One would expect that the cost of adaptive schedules decreases monotonically in the rescheduling frequency (i.e., increases monotonically in the rescheduling time Δ). While this expectation generally holds true, we start by presenting an elementary instance in which this is *not* the case.

We first consider a situation of three clients with exponentially distributed service times, where we normalize the mean service time to 1. The number of clients n is equal to 3, and the weight ω is equal to 0.5 (i.e., the idle and waiting times are of equal importance). Note that in this setting with the SCV of the service times being equal to 1, due to the memoryless property of the exponential distribution, there is no impact of the elapsed service time. As a result, the model can (to some extent) be analyzed explicitly. In Fig. 3, we assess the influence of the rescheduling time Δ on the cost by simulation. It is observed that in this instance, the cost is not monotonically increasing. Also, for various values of Δ there are discontinuities. An extensive explicit analysis of this counterintuitive behavior can be found in Appendix A.

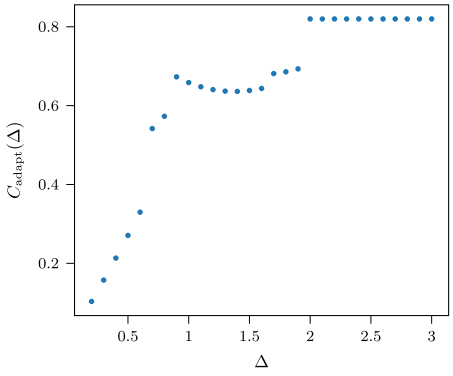


Fig. 3. Cost of adaptive schedule for $n = 3$ and $\omega = 0.5$.

Now we focus, in instances with a more realistic number of clients n, on the impact of the weight ω (i.e., the importance of the idle time relative to the waiting time). In Table 1, we consider three values of the weight ω. The cost of static scheduling corresponds to the instance $\Delta = \infty$. The main conclusion from the table is that, in particular for low Δ and ω there is a strong cost reduction. Even for $\Delta = 8$, the efficiency gain is still around 10%, which is significant considering that in most cases, the schedule will be updated only once (since there are fifteen clients).

Additionally, it can be seen that both the sum of the idle times and the sum of the waiting times substantially decrease each time we halve the rescheduling time. We also observe that increasing the rescheduling frequency has more impact on the waiting time than on the idle time. When the waiting time is of higher importance than idle time, i.e., for low values of the weight ω, the waiting time of each of the clients can

Table 1
Cost and efficiency gain of adaptive schedule, $n = 15$.

ω	Δ	0.5	1	2	4	8	∞
0.2	$\sum \mathbb{E}I_i$	3.79	8.15	13.08	16.11	15.72	16.95
	$\sum \mathbb{E}W_i$	0.00	0.00	1.85	1.41	1.94	2.42
	$C_{\text{adapt}}(\Delta)$	0.76	1.63	4.10	4.35	4.69	5.33
	$\Gamma(\Delta)$	85.8%	69.4%	23.1%	18.4%	11.9%	0.0%
0.5	$\sum \mathbb{E}I_i$	3.79	6.60	6.06	7.35	7.40	8.14
	$\sum \mathbb{E}W_i$	0.00	2.09	3.71	5.06	6.13	6.95
	$C_{\text{adapt}}(\Delta)$	1.89	4.35	4.89	6.20	6.76	7.55
	$\Gamma(\Delta)$	74.9%	42.4%	35.2%	17.8%	10.4%	0.0%
0.8	$\sum \mathbb{E}I_i$	1.05	1.81	2.15	2.47	2.76	2.98
	$\sum \mathbb{E}W_i$	8.30	7.64	10.52	13.50	15.40	17.32
	$C_{\text{adapt}}(\Delta)$	2.50	2.97	3.82	4.68	5.29	5.85
	$\Gamma(\Delta)$	57.3%	49.2%	34.7%	20.1%	9.6%	0.0%

even be completely eliminated by rescheduling at a high frequency. The benefit of frequent rescheduling (i.e., the gain) is more noticeable when the relative importance of waiting time is higher. In those settings rescheduling tends to avoid excessive waiting times, i.e., it will be less likely that many clients are waiting in the system. In Fig. 4 we observe that for any fixed rescheduling length Δ, the total cost of the schedule depends effectively linearly on the number of clients to schedule.

Then, we examine the dependence of the number of times the schedule gets updated and the cost of the schedule on the rescheduling length Δ, for various values of the weight ω. In Fig. 5 we observe for homogeneous exponentially distributed service times that the number of updates increases sharply as Δ decreases. The higher the importance of the waiting times (i.e., the lower ω), the more often the schedule will be updated, as can be reasoned as follows. When the weight assigned to waiting time (i.e., $1 - \omega$) increases, each appointment is given a longer time interval. This means that the time it takes to execute all appointments (i.e., the *makespan*) goes up. As a result, the number of schedule updates increases.

Table 1 has shown that, generally, increasing the rescheduling frequency (i.e., decreasing Δ) leads to a reduction in both the sum of idle times and the sum of waiting times. However, this does not provide any insight into the effect on the *individual* idle and waiting times. To examine this behavior in greater detail, Figs. 6 and 7 present the mean values of idle and waiting time for each client, respectively, for $\omega = 0.2$ and $\omega = 0.8$.

- In line with e.g., Kuiper (2016, Figure 2.9.b), we note that in the static case ($\Delta = \infty$), the mean idle times for clients exhibit a dome shape, mimicking the optimal individual interarrival times. Conversely, the mean waiting time increases with a client's position, i.e., clients scheduled later are expected to experience longer waiting times; cf. Kuiper (2016, Figure 2.10.b).
- Overall, updating the schedule leads to shorter idle and waiting times for future clients. This effect is particularly visible for $\omega = 0.8$, as clients after the first rescheduling moment have significantly lower mean idle and waiting times compared to the static case. For $\omega = 0.2$, although the average idle and waiting times are generally lower than in the static case, a zigzag pattern emerges in the mean idle and waiting times of future arrivals. This can be attributed to 'asynchronizations', where the rescheduling period does not align with the arrival of a client: when rescheduling occurs just before a client's arrival, the average idle time is more likely to increase while the waiting time for this client is likely to decrease, still resulting in notable cost savings for that specific client. This zigzag effect becomes less pronounced as the rescheduling frequency increases, eventually leading to a relatively constant average idle and waiting time for all clients.

Fig. 8 shows the influence of the rescheduling time on the cost for $\omega = 0.2$ and $\omega = 0.8$. While we again keep the mean service times equal to 1, we now also vary the SCV. It is observed that the

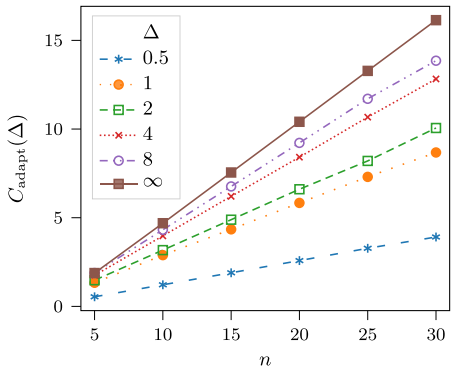


Fig. 4. Cost of adaptive schedule for $\omega = 0.5$ and different values of n and Δ.

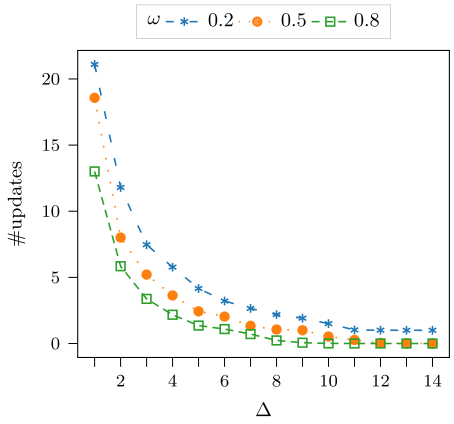


Fig. 5. Number of updates of adaptive schedule for $n = 15$.

cost is increasing in $\mathbb{S}(B)$, which is in line with the findings of Mahes et al. (2023). We have connected the dots, but clearly, between the data points discontinuities can occur. While the figures show that at a more global level increasing the scheduling frequency leads to a cost reduction, it also displays the local non-monotone behavior that we already encountered in the elementary instance above. We in particular observe that for low values of ω, the cost locally exhibits sharply decreasing behavior. As shown in the appendix, the observed peaks (in particular for high SCV values and $\omega = 0.2$) are located at the scheduled arrival times. In fact, the cost function is not continuous and makes upward jumps at $t_1, t_2, \ldots$. Therefore, it is better to choose an update interval Δ such that updates do not coincide with arrival epochs.

Additionally, the impact of the service time variability on the efficiency gain is worth considering. Our experiments reveal that higher values of $\mathbb{E}B$ or $\mathbb{S}(B)$ correspond not only to higher cost but also to larger efficiency gains. The gain being increasing in $\mathbb{E}B$ can be explained by the fact that longer service times lead to more rescheduling opportunities, assuming a fixed rescheduling length Δ. The gain is also increasing in $\mathbb{S}(B)$: if there is a higher variability in the service times, the effect of relatively extreme scenarios (i.e., many or no clients in the system) can be neutralized more effectively when scheduling adaptively.

Experiment 2 (*Heterogeneous Service Times*)**.** While in the previous experiment, we worked with homogeneous service times, in this experiment we assess the impact of heterogeneity. First of all, we do this in the setting of exponentially distributed service times (i.e., $\mathbb{S}(B_i) = 1$), with different parameters (and $\Delta = 3$ and $\omega = 0.5$ fixed). We consider a situation in which the service times are ordered such that the mean service times $\mathbb{E}B_i \in \{0.5, 0.6, \ldots, 1.5\}$ (and hence the corresponding variances) are increasing, three in which the service times are permuted

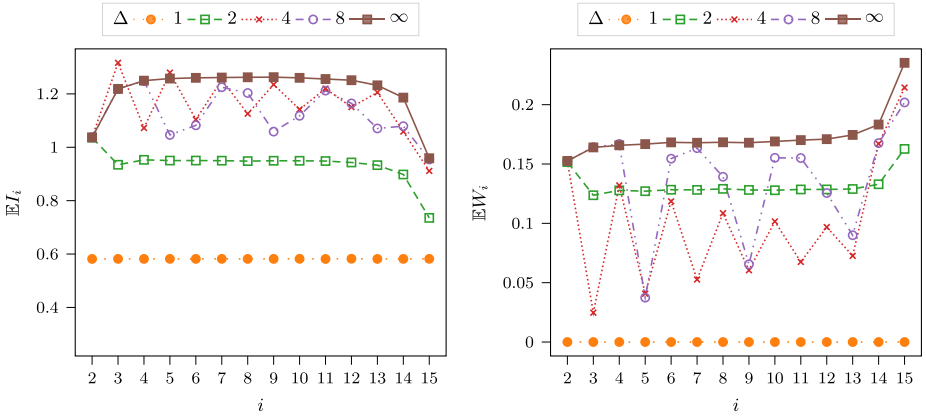


Fig. 6. Mean idle and waiting time of each client $i = 2, \ldots, n$ for $\omega = 0.2$, $n = 15$, and different values of Δ.

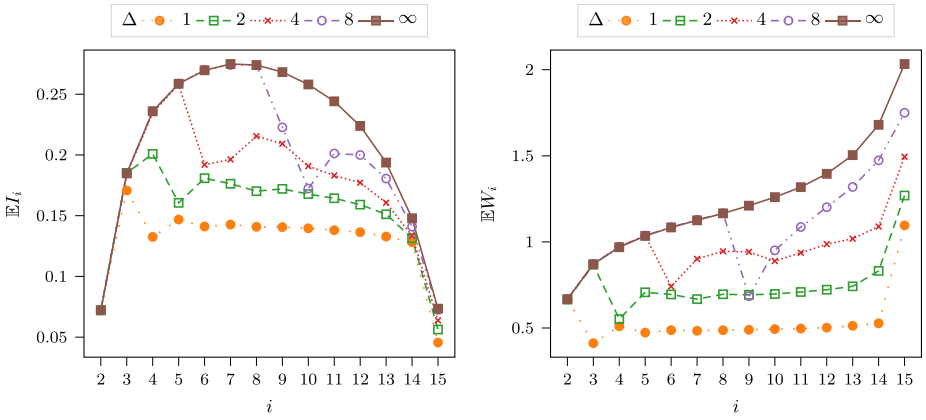


Fig. 7. Mean idle and waiting time of each client $i = 2, \ldots, n$ for $\omega = 0.8$, $n = 15$, and different values of Δ.

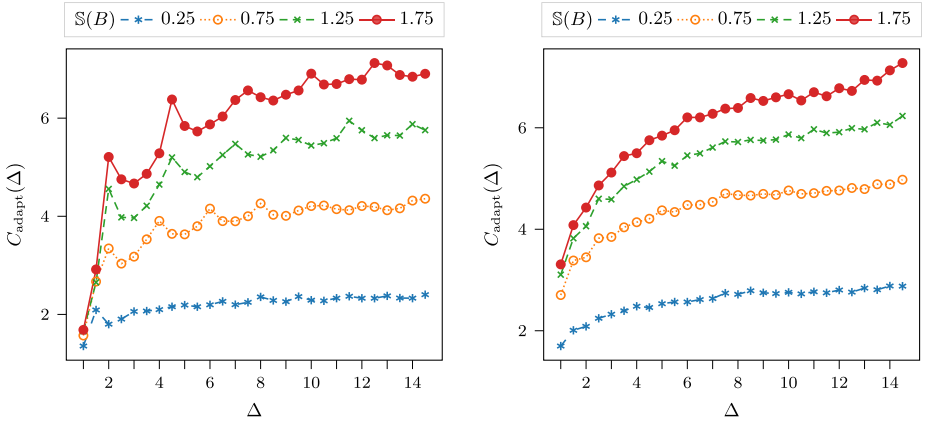


Fig. 8. Cost of adaptive schedule for $n = 15$ and $\omega = 0.2$ (left) and $\omega = 0.8$ (right).

randomly, and one in which the mean service times are ordered decreasingly. The results are depicted in Table 2. The experiments confirm that the cost is lower in the case of increasing mean service times (and hence increasing variances), in line with the findings of de Kemp et al. (2021) and Kong et al. (2016). Secondly, we keep the $\mathbb{E}B_i = 1$ fixed, but take the $\mathbb{S}(B_i) \in \{0.5, 0.6, \ldots, 1.5\}$. Again, from Table 2 it is observed that the cost is the lowest when the variances of the jobs are increasing. These conclusions are particularly relevant in applications where the service-time distributions can vary substantially, confirming that clients with short (expected) service times should be scheduled first.

4.3. More practical variants

In reality, there are good reasons to keep the adapted schedule relatively close to the existing one. Clients may be on their way to the service facility, for instance, or may negatively perceive frequent schedule updates. In this subsection, three mechanisms that deal with such considerations are proposed and evaluated. Each case starts with the static schedule that is obtained by solving optimization problem (1). Then, the schedule is updated using a policy based on the minimization of a different objective function. Finally, to assess the cost of the adaptive schedules we use simulation, where in each simulation run the weighted sum of the realized idle and waiting times is determined. As

Table 2
Cost of adaptive schedule for $n = 11$, $\omega = 0.5$ and $\Delta = 3$. In the left table, we take $\mathbb{E}B_i \in \{0.5, 0.6, \ldots, 1.5\}$ equidistant, while in the right table the $\mathbb{S}(B_i) \in \{0.5, 0.6, \ldots, 1.5\}$ are varied in the same manner.

$\mathbb{S}(B_i) = 1$		$\mathbb{E}B_i = 1$	
$\mathbb{E}B_i$	$C_{\text{adapt}}(\Delta)$	$\mathbb{S}(B_i)$	$C_{\text{adapt}}(\Delta)$
Increasing	3.82	Increasing	3.76
Permutation 1	3.97	Permutation 1	3.88
Permutation 2	4.25	Permutation 2	4.02
Permutation 3	4.16	Permutation 3	4.02
Decreasing	4.26	Decreasing	4.08

before, when evaluating the cost of adaptive schedules we use $N = 10^6$ simulation runs, unless otherwise stated.

Experiment 3 (*Optimizing Δ by Penalizing Adaptations*)**.** This experiment discusses a way to determine a suitable update interval Δ. Clearly, in practical situations, there are various reasons why one would wish to avoid frequent adaptations. One could incorporate this in various ways, for instance by penalizing the number of adaptations. A possible way to achieve this is by choosing the rescheduling time Δ such that, for some given weight $\alpha \in [0, 1]$, the new cost function

$$C^{\star}_{\text{adapt}}(\Delta \mid \alpha) := \alpha\, C_{\text{adapt}}(\Delta) + (1 - \alpha)\frac{1}{\Delta}$$

is minimized. In our experiment, we consider the setting in which the service times are independent identically distributed exponential random variables, and the weight $\omega = 0.5$ is held fixed. We are interested in which Δ minimizes the considered cost function for various values of α. The results are shown in Fig. 9.

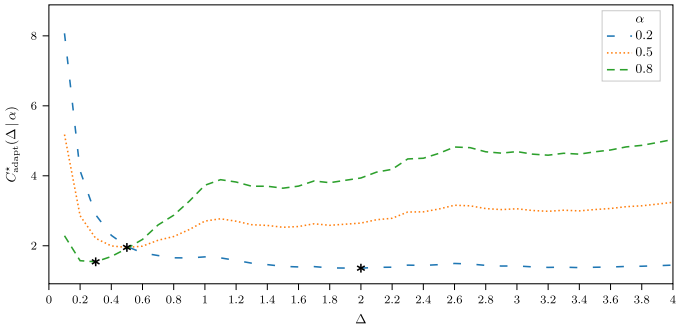


Fig. 9. Cost of adaptive schedule for different values of Δ for $n = 15$ clients, $\omega = 0.5$, and $\mathbb{E}B = 1$ and $\mathbb{S}(B) = 1$. The simulations are each based on $N = 10^3$ runs.

It is observed that the cost function is relatively flat around its minimum. More importantly, a lower weight α represents a more substantial penalization for adaptations, and thus results in a higher optimal rescheduling time, as expected.

Experiment 4 (*Tradeoff with Fixed Interval*)**.** In this experiment, we consider the variant in which the appointment times within a time interval of length τ after each of the rescheduling epochs are held fixed. This means that only the clients that are supposed to arrive at least τ time units later can undergo schedule updates. To make this variant even more realistic, we also impose that these clients will still arrive at least τ time units later. We call the cost $C^{\circ}_{\text{adapt}}(\Delta \mid \tau)$. We again focus on homogeneous exponential service times, with the weight $\omega = 0.5$ being fixed. In Fig. 10 we present the combination of Δ and τ that leads to the same cost, so as to obtain insight into the tradeoff between these two parameters. The figure confirms an evident property: the higher the fixed schedule time length τ, the lower the rescheduling time Δ should be in order to achieve the same cost.

In Table 3, we present the gain $\Gamma(\Delta \mid \tau)$ achieved by adaptive scheduling with a fixed schedule time length $\tau \in \{0, 1, 2\}$ over static scheduling; this gain is quantified analogously to Eq. (2). Note that

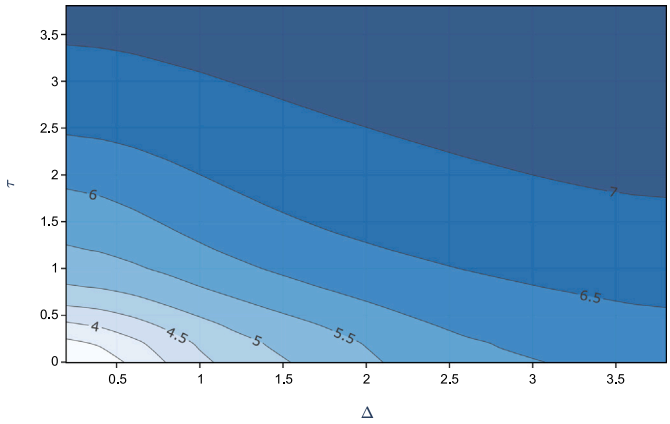


Fig. 10. Contour plot of $C^{\circ}_{\text{adapt}}(\Delta \mid \tau)$ for $n = 15$ and $\omega = 0.5$. The simulations are each based on $N = 2500$ runs.

Table 3
Efficiency gain $\Gamma(\Delta \mid \tau)$ of adaptive schedule, $n = 15$.

ω	Δ	0.5	1	2	4	8	∞
0.2	$\tau = 0$	85.8%	69.4%	23.1%	18.4%	11.9%	0.0%
	$\tau = 1$	33.2%	17.0%	14.4%	6.3%	4.0%	0.0%
	$\tau = 2$	12.9%	11.2%	7.0%	4.3%	2.0%	0.0%
0.5	$\tau = 0$	74.9%	42.4%	35.2%	17.8%	10.4%	0.0%
	$\tau = 1$	29.1%	23.7%	14.9%	9.8%	4.9%	0.0%
	$\tau = 2$	14.7%	13.3%	9.3%	5.5%	3.4%	0.0%
0.8	$\tau = 0$	57.3%	49.2%	34.7%	20.1%	9.6%	0.0%
	$\tau = 1$	31.5%	28.0%	19.9%	11.5%	5.6%	0.0%
	$\tau = 2$	17.9%	17.4%	12.2%	8.5%	3.8%	0.0%

$\tau = 0$ corresponds with the standard variant of adaptive scheduling as considered in Experiment 1. We conclude from the table that the gain achieved can be substantial. At the same time, for evident reasons, choosing τ relatively large rules out a strong cost reduction (compared to static scheduling, that is).

Experiment 5 (*Penalizing Deviations*)**.** We then consider a variant in which deviations relative to the previous schedule are penalized. Concretely, in self-evident notation, at any adaptation, we minimize the objective function

$$h(t_1^{\text{new}}, \ldots, t_n^{\text{new}}; t_1^{\text{old}}, \ldots, t_n^{\text{old}} \mid k, u) := \alpha\, f(t_1^{\text{new}}, \ldots, t_n^{\text{new}} \mid k, u) + (1 - \alpha) \sum_{i=1}^{n} (t_i^{\text{new}} - t_i^{\text{old}})^2$$

for some weight $\alpha \in [0, 1]$. Working with homogeneous exponentially distributed service times, Fig. 11 expresses the dependency of the objective function on the weight α.

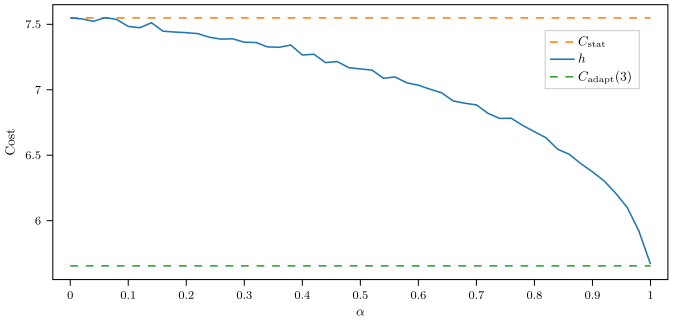


Fig. 11. Cost of adaptive scheduling using objective function h for $n = 15$ clients with weight $\omega = 0.5$ and $\Delta = 3$. The simulations are based on $N = 50,000$ runs.

The case $\alpha = 0$ corresponds to penalizing any deviation from the initial static schedule and thus leads to the same cost as of the static schedule, i.e., C_{stat}. The higher α, the deviations relative to the previous schedule are penalized to a lesser extent, leading to more flexible schedules and thus lower cost. We see that the cost when using objective function h gradually decreases to the scenario of rescheduling after each $\Delta = 3$ time units without penalizing any deviation from the old schedule. A reasonably smooth curve is already being achieved for $N = 50,000$ simulation runs.

Experiment 6 (*Including Overtime*)**.** Various extensions are possible, for instance, the one in which overtime is penalized. Overtime, for a given horizon $T > 0$, is defined as

$$O_T := \max\left\{\sum_{i=1}^{n} B_i + \sum_{i=1}^{n} I_i - T, 0\right\},$$

where $\sum_{i=1}^{n} B_i + \sum_{i=1}^{n} I_i = t_n + S_n$ is to be interpreted as the *session end time*. In this setup, the problem of finding an optimal schedule could include a penalty for the amount of time the horizon T is exceeded. This concretely means that the objective function becomes, for a weight $\beta > 0$,

$$\omega \sum_{i=1}^{n} \mathbb{E}I_i + (1-\omega) \sum_{i=1}^{n} \mathbb{E}W_i + \beta\, \mathbb{E}O_T.$$

The overtime being increasing in t_n, the new term in the objective function gives an extra incentive to have low idle times. The term $\mathbb{E}O_T$ can be evaluated using the machinery presented above:

$$\mathbb{E}O_T = \mathbb{E}\max\{t_n + S_n - T, 0\} = \int_0^\infty \mathbb{P}(S_n \geqslant t - t_n + T)\, \mathrm{d}t,$$

which in the terminology of Proposition 1 equals

$$\mathbb{E}O_T = \max\{t_n - T, 0\} - \boldsymbol{G}_n \exp(\boldsymbol{V}_n \max\{T - t_n, 0\})\, \boldsymbol{V}_n^{-1} \mathbf{1}_{D_n}.$$

Also for this type of objective function, one could work with an adaptive scheduling approach, so as to reduce the objective function.

To demonstrate the impact of including overtime, Fig. 12 illustrates the relationship between the cost and the horizon T. The experiment retains the same configuration as the previous one. As anticipated, the objective function is decreasing in the horizon T: the higher the value of T, the lower the overtime, and hence the lower the cost. Setting a high horizon T essentially corresponds to not imposing any overtime constraint, resulting in a cost function that disregards the overtime component (i.e., resulting in $\beta = 0$). Our numerical output shows that the cost of the adaptive schedules appears to behave similarly to that of the static schedule, with the absolute cost difference between the two showing an almost negligible dependency on the horizon T. In the example, for small values of T, adaptive scheduling leads to a 12% cost reduction, whereas for large T, this cost reduction is as high as 28%.

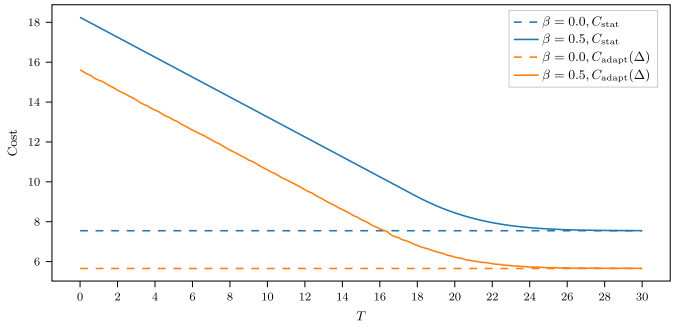


Fig. 12. Cost of static and adaptive scheduling with horizon T for $n = 15$ clients with weight $\omega = 0.5$ and $\Delta = 3$. The simulations are based on $N = 10^5$ runs.

Table 4
Cost of adaptive and static schedule when rescheduling at each start of service, $n = 15$ and $\mathbb{E}B = 1$.

ω	$\mathbb{S}(B)$	0.25	0.5	0.75	1.0	1.25	1.5	1.75
0.2	C_{adapt}	2.22	3.29	4.14	4.94	5.68	6.36	7.00
	C_{stat}	2.41	3.57	4.46	5.33	6.18	6.94	7.64
	Γ	7.9%	7.8%	7.2%	7.2%	8.0%	8.3%	8.4%
0.5	C_{adapt}	2.92	4.24	5.31	6.20	6.97	7.68	8.34
	C_{stat}	3.61	5.22	6.45	7.55	8.49	9.33	10.09
	Γ	19.1%	18.8%	17.7%	17.8%	17.9%	17.7%	17.3%
0.8	C_{adapt}	1.94	2.81	3.55	4.14	4.66	5.14	5.58
	C_{stat}	2.96	4.18	5.11	5.85	6.40	6.88	7.31
	Γ	34.4%	32.8%	30.5%	29.3%	27.1%	25.3%	23.6%

4.4. Comparison with alternative rescheduling methods

In this subsection, we compare our adaptive approach with a number of alternative rescheduling schemes. We limit ourselves to rescheduling only on moments when the system is not empty, making it unlikely for the next client to be scheduled to show up right away (as this would enforce immediate waiting time).

Experiment 7 (*Rescheduling at Start of Service*)**.** First, we consider a strategy in which one reschedules at every moment when a client's service *starts*. The results are presented in Table 4, where we vary the value of the weight ω and the SCV. Here, Γ is the gain achieved by rescheduling as defined in Eq. (2). It is observed that in all cases, substantial gains can be achieved. For a small value of ω, the gain is roughly constant, but a higher gain is achieved for SCVs further away from 1. The higher the relative importance of the idle time, the more can be gained by scheduling adaptively, in particular for small values of the SCV.

Experiment 8 (*Rescheduling at Arrivals*)**.** Suppose that one reschedules at every moment that a client enters the system. This is in principle the same mechanism as the one analyzed in Mahes et al. (2023), with the crucial difference being that the dynamic programming approach 'exploits' the fact that when rescheduling one knows that there will be more future rescheduling moments (so that the mechanism analyzed in Mahes et al. (2023) necessarily leads to lower cost). While using this knowledge yields truly optimal results, the enormous state space makes real-time computations of schedules using the dynamic programming approach computationally challenging. This experiment serves to quantify the difference between both algorithms.

Denote by C_{dyn} the cost of the dynamic programming method studied in Mahes et al. (2023). Then we define the loss of scheduling using the proposed method, rather than scheduling following the dynamic programming method of Mahes et al. (2023), by

$$\ell := \frac{C_{\text{adapt}} - C_{\text{dyn}}}{C_{\text{dyn}}}.$$

Also, let Γ be the gain achieved by rescheduling. Find in Table 5 the results for different values of ω and SCV. It is observed that scheduling adaptively at each client arrival leads to nearly the same cost as the theoretical optimum achieved by scheduling as in Mahes et al. (2023). More specifically, the loss obtained by the proposed method is always less than 3%, and therefore, can be considered negligible. This is good news: when using our computationally inexpensive adaptive approach, instead of the computationally heavy dynamic programming approach of Mahes et al. (2023), hardly any efficiency is lost. In line with the findings reported in Mahes et al. (2023), the gains achieved by rescheduling at each client arrival are always substantial and more pronounced when the values of ω and the SCV are high.

Table 5
Cost of adaptive and dynamic schedule when rescheduling at each client arrival, $n = 15$ and $\mathbb{E}B = 1$. The simulations are based on $N = 10^6$ runs.

ω	$\mathbb{S}(B)$	0.25	0.5	0.75	1.0	1.25	1.5	1.75
0.2	C_{adapt}	2.27	3.33	4.13	4.86	5.45	5.96	6.41
	C_{dyn}	2.26	3.31	4.11	4.83	5.39	5.87	6.29
	ℓ	0.6%	0.5%	0.6%	0.6%	1.1%	1.5%	1.8%
	Γ	5.6%	6.9%	7.2%	8.9%	11.8%	14.1%	16.1%
0.5	C_{adapt}	3.12	4.40	5.40	6.15	6.68	7.15	7.56
	C_{dyn}	3.07	4.34	5.32	6.05	6.55	6.97	7.35
	ℓ	1.7%	1.5%	1.4%	1.6%	2.0%	2.5%	2.8%
	Γ	13.6%	15.7%	16.4%	18.6%	21.2%	23.4%	25.1%
0.8	C_{adapt}	2.20	3.05	3.71	4.14	4.43	4.68	4.89
	C_{dyn}	2.16	2.99	3.64	4.07	4.33	4.56	4.76
	ℓ	2.7%	2.1%	1.9%	1.7%	2.2%	2.4%	2.6%
	Γ	25.1%	27.0%	27.5%	29.2%	30.8%	32.1%	33.1%

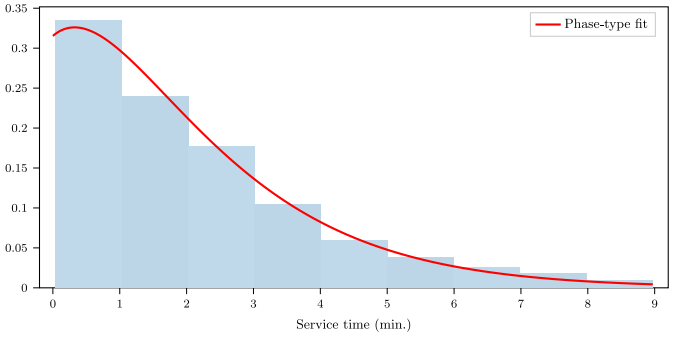


Fig. 13. Service-time distribution and corresponding phase-type fit.

4.5. Using real-world data

The following experiment provides an illustrative example of our method's performance for a real-world data set.

Experiment 9 (*Scheduling Arrivals for Parcel Delivery*)**.** The data available consists of (anonymized versions of) 60 traces, stemming from a parcel delivery company. A trace corresponds to a route along which a driver has delivered parcels. Using the first 50 traces we have estimated the mean and variance of the service-time distribution (corresponding to the travel time between two subsequent customers, increased by the time required to hand over the parcel). Then our algorithm is run on the last 10 traces, i.e., the traces not used in the estimation.

Across the first 50 routes, a total of 4,930 parcels have been successfully delivered. The service times associated with these deliveries have an estimated mean of 2.152 min. and an estimated squared coefficient of variation (SCV) of 0.738. Then the two-moment fit outlined in Section 3.1 is employed. As we have an SCV smaller than 1, the weighted Erlang distribution is used. When fitting the mean and SCV, we arrive at the parameters $K = 1$, $\mu = 0.728$, and $p = 0.433$. Fig. 13 presents a histogram of the distribution of the service times, accompanied by the corresponding phase-type fit. The per-route number of parcels to be delivered varies between 33 and 129, with an average of 98.6 parcels per route.

Once the parameters of the service-time distribution have been determined based on the data from the initial 50 routes, we can generate schedules for the subsequent 10 routes. In our example, we have worked with a weight parameter of $\omega = 0.5$ (signifying equal importance assigned to idle and waiting times). The schedules are updated every $\Delta = 15$ min. A comparison is made between the cost incurred on these routes using the adaptive approach on one hand, and the cost if one would not do any schedule updates on the other hand. The outcome of this comparison is presented in Table 6.

The adaptive scheduling approach consistently leads to a substantial reduction of the overall cost on all 10 routes, with an average gain of over 10%. By adapting the schedule for the rides that consist of an average of 101.5 parcels, approximately 21 updates are made, indicating that the schedule is adjusted after delivering roughly every 5 parcels. The reduction in waiting times shows a similar relative decrease.

Intentionally, we kept this experiment as elementary as possible; the main goal was to show our approach's potential to reduce the cost significantly. In our setup, all traces were 'treated equally', in that it was implicitly assumed that service times stem from the same distribution, but one can further refine the approach by classifying routes based on specific features (weather, driver, etc.). In that case, one estimates the mean and SCV of the per-class service times and uses these when generating schedules.

5. Concluding remarks

In this work, we have introduced a methodology to generate appointment schedule updates at any time. Given the first two moments of the clients' service times, a phase-type fit is applied, thus facilitating a fast evaluation of the objective function, which is a weighted combination of mean idle times and mean waiting times. In the update procedure that we propose, the cost function takes into account the current state information, namely the number of clients in the system as well as the elapsed service time of the client in service (if any). The evaluation of the cost function that uses this state information, as well as its optimization over the appointment times, have the same complexity as their counterparts pertaining to the conventional, static case.

Our results include an applet that can be used immediately to generate and update schedules. Experiments show that updating can lead to substantially lower cost, even if we do not update very frequently and/or leave the first appointments in the current schedule unchanged.

We envisage various directions for follow-up research. First, the cost function can be modified in various ways, a natural extension being a cost function that incorporates appointment *windows* rather than appointment times. This can be useful, for example, in the context of a service in which clients are served at home (such as home delivery services). While in the present paper we have worked with mean idle and waiting times, other functional forms are worth studying, too; one can for instance consider a quadratic cost function to penalize large outcomes more strongly.

Second, in this paper, we focused predominantly on updating the schedule periodically, even though with the proposed framework a schedule could in principle be updated at any time. One can therefore think about the question *when* one should update (given the available state information), for instance, if there is a maximum on the number of updates.

CRediT authorship contribution statement

Roshan Mahes: Conceptualization, Methodology, Software, Writing – original draft. **Michel Mandjes:** Conceptualization, Methodology, Writing – original draft. **Marko Boon:** Conceptualization, Methodology, Software, Writing – original draft.

Data availability

The data used in Experiment 9 is available at https://adaptiveschedule.eu.pythonanywhere.com/download/service_times.csv.

Table 6
Realized sums of waiting times and total cost of static and adaptive schedule (in minutes) for parcel delivery. The bottom row displays the reduction (percentagewise) of the objective function.

	Day	51	52	53	54	55	56	57	58	59	60	Mean
	n	115	103	104	96	84	86	84	116	115	112	101.5
Static	$\sum_{i=1}^{n} W_i$	161.82	35.92	61.39	112.71	12.68	71.55	87.16	29.77	53.05	37.36	66.34
	C_{stat}	146.34	90.07	92.02	94.45	94.16	78.24	94.72	115.21	131.50	99.54	103.62
Adaptive	#updates	25	21	22	21	16	18	17	24	23	23	21
	$\sum_{i=1}^{n} W_i$	116.73	31.14	63.90	106.16	10.80	62.36	77.23	34.09	53.78	38.15	59.43
	$C_{\text{adapt}}(\Delta)$	121.34	78.77	85.65	92.65	79.13	69.56	84.80	104.33	115.95	89.04	92.12
Gain	$\Gamma(\Delta)$	17.1%	12.5%	6.9%	1.9%	16.0%	11.1%	10.5%	9.4%	11.8%	10.5%	11.1%

Appendix A. Optimizing the objective function: Special cases

In Section 3.3 we have described how to optimize the objective function, so as to determine the optimal schedule. In Section 4, several experiments are performed to assess the influence of each of the parameters on the cost of the schedule. In Experiment 1, it is observed that the shape of the cost as a function of Δ, the length of the update interval, is rather complex. In particular, a remarkable conclusion is that, counterintuitively, the cost is not monotone in Δ. To explain this behavior, we analyze three relatively small systems with homogeneous exponential service times with means $\mathbb{E}[B_i] = 1$ probabilistically, the last of which corresponds to the instance featured in Experiment 1.

Case 1: $n = 2,\ k = 0$

The first case we study, is the simplest possible case, with $n = 2$ clients. Although we start with an empty system ($k = 0$), the first client is always scheduled at time $t_1 = 0$, i.e., this case is the same as $n = 2$, $k = 1$. There remains only one more client to be scheduled, at time t_2. Since $I_1 = W_1 = 0$, the objective function simplifies to

$$f(t_1, t_2 \mid k = 0) = \omega\,\mathbb{E}[I_2] + (1-\omega)\,\mathbb{E}[W_2].$$

In this setup, we can omit the variable u, used to indicate the elapsed service time, because it is irrelevant due to the memorylessness of the service times. We consider the standard variant, as described in Experiment 1, where we periodically update the schedule each Δ time units. The corresponding cost of the adaptive schedule will be denoted by $C^{n,k}_{\text{adapt}}(\Delta)$. First, we consider the case $\Delta \to \infty$, meaning that we never update the schedule. For this simple case, we can explicitly compute the expected idle times and waiting times:

$$\mathbb{E}[I_2] = \int_0^{t_2} (t_2 - t)e^{-t}\mathrm{d}t = t_2 - 1 + e^{-t_2},$$
$$\mathbb{E}[W_2] = \int_{t_2}^{\infty} (t - t_2)e^{-t}\mathrm{d}t = e^{-t_2},$$

so that the cost is given by

$$f(t_1, t_2 \mid k = 0) = \omega(t_2 - 1) + e^{-t_2}.$$

This expression is minimized for $t_2 = t_2^{\text{opt}} := -\log\omega$, resulting in cost $-\omega\log\omega$.

When analyzing the adaptive scheduling policy when Δ is finite, we first observe that when $\Delta \geqslant t_2^{\text{opt}}$, the first possibility to update the schedule takes place only *after* the second (and last) client has arrived. This implies that there are no more clients left to schedule and that the cost will simply be equal to those when $\Delta = \infty$.

A more interesting situation arises when $\Delta < t_2^{\text{opt}}$, because now the cost depends on whether or not the service time B_1 has elapsed. Let us first consider the case where the service time B_1 has elapsed at time $t = \Delta$, which happens with probability

$$\mathbb{P}(B_1 < \Delta) = 1 - e^{-\Delta}.$$

In this case, the idle time is equal to $\Delta - B_1$ with expectation

$$\mathbb{E}[I_2 \mid B_1 < \Delta] = \frac{\mathbb{E}[(\Delta - B_1)\mathbf{1}_{\{B_1<\Delta\}}]}{\mathbb{P}(B_1 < \Delta)} = \frac{1}{1-e^{-\Delta}}\int_0^{\Delta}(\Delta - t)e^{-t}\mathrm{d}t$$

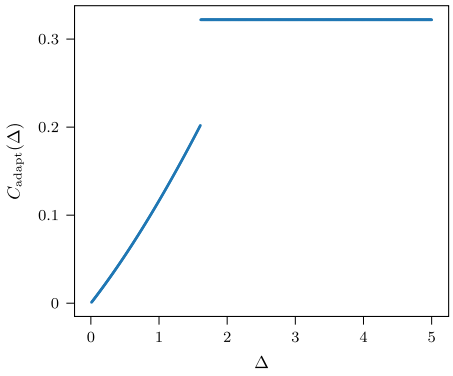


Fig. 14. Cost versus Δ for $n = 2$, $k = 0$ and $\omega = 0.2$ with homogeneous exponential service times.

$$= \frac{\Delta + e^{-\Delta} - 1}{1 - e^{-\Delta}}.$$

In the case where the service time B_1 has *not* elapsed at time $t = \Delta$, which happens with probability $e^{-\Delta}$, due to the memoryless property of the exponential distribution, the remaining cost is equal in distribution to the cost at $t = 0$. This gives us the following equation for the total cost when $\Delta < t_2^{\text{opt}}$:

$$\mathbb{E}[C^{2,0}_{\text{adapt}}(\Delta)] = \omega\,\mathbb{P}(B_1 < \Delta)\mathbb{E}[I_2 \mid B_1 < \Delta] + \mathbb{P}(B_1 > \Delta)\,\mathbb{E}[C^{2,0}_{\text{adapt}}(\Delta)].$$

Solving this equation, finally, leads to the following elegant expression for the total cost:

$$\mathbb{E}[C^{2,0}_{\text{adapt}}(\Delta)] = \begin{cases} \omega\dfrac{e^{\Delta}(1-\Delta) - 1}{1 - e^{\Delta}}, & \Delta < -\log\omega, \\ -\omega\log\omega, & \Delta \geqslant -\log\omega. \end{cases} \tag{3}$$

In Fig. 14 we have shown the cost for $\omega = 0.2$. It can clearly be seen that the cost increases until $\Delta = t_2^{\text{opt}} = 1.609$. For larger Δ it remains constant at the level $C_{\text{adapt}}(\Delta) = C_{\text{adapt}}(\infty) = 0.322$.

Case 2: $n = 3,\ k = 2$

The next example concerns a system with three clients, two of which are already present at time zero: $t_1 = t_2 = 0$. Only the third client needs to be scheduled. Note that the waiting time of client 2 starts immediately at time $t = 0$ and will always equal B_1, the service time of the first client. Using Lindley's recursion, we find

$$I_1 = W_1 = I_2 = 0,\ \ W_2 = B_1,\ \ I_3 = \max\{t_3 - B_1 - B_2, 0\},$$
$$W_3 = \max\{B_1 + B_2 - t_3, 0\}. \tag{4}$$

The objective function is equal to

$$f(t_1, t_2, t_3 \mid k = 2) = \omega\,\mathbb{E}[I_3] + (1-\omega)\,(\mathbb{E}[W_2] + \mathbb{E}[W_3]). \tag{5}$$

Substitution of Eq. (4) in Eq. (5) gives, after deconditioning,

$$f(t_1, t_2, t_3 \mid k = 2) = (t_3 - 3)\,\omega + e^{-t_3}(t_3 + 2) + 1.$$

This function is minimized for $t_3 = t_3^{\text{opt}} = -\mathcal{W}_{-1}(-\omega/e)-1$, where $\mathcal{W}_{-1}(\cdot)$ denotes the branch at -1 of the Lambert-W function. For $\omega = 0.2$, the numerical value is equal to $t_3^{\text{opt}} = 2.994$ with corresponding cost $\mathbb{E}[C^{3,2}_{\text{adapt}}(\infty)] = 1.249$. As before, we note that

$$\mathbb{E}[C^{3,2}_{\text{adapt}}(\Delta)] = \mathbb{E}[C^{3,2}_{\text{adapt}}(\infty)] \text{ for } \Delta \geqslant t_3^{\text{opt}},$$

which we refer to as Case 1.

Case 2 (i.e., $\Delta < t_3^{\text{opt}}$) is now more complex, since there are *three* subcases that we need to distinguish:

(2a) The service times $B_1 + B_2$ have elapsed at time $t = \Delta$. This happens with probability

$$\mathbb{P}(B_1 + B_2 < \Delta) = 1 - (1+\Delta)e^{-\Delta},$$

which is simply the probability that an Erlang(2,1) random variable is less than Δ. In this subcase, the idle time is $I_3 = \Delta - B_1 - B_2$ and the third client should enter immediately. Since we also have cost related to the waiting time W_2, the total cost is

$$\begin{aligned}
&\mathbb{E}[C^{3,2}_{\text{adapt}}(\Delta) \mid B_1 + B_2 < \Delta] \\
&\quad = \frac{1}{1-(1+\Delta)e^{-\Delta}} \\
&\qquad \times \int_0^{\Delta} \int_0^{\Delta - b_1} \big(\omega(\Delta - b_1 - b_2) + (1-\omega)b_1\big) e^{-b_1} e^{-b_2} \,\mathrm{d}b_2 \,\mathrm{d}b_1 \\
&\quad = \frac{1 + (\Delta - 3)\omega + e^{-\Delta}\big((3+2\Delta)\omega + (\omega - 1)\frac{\Delta^2}{2} - 1 - \Delta\big)}{1-(1+\Delta)e^{-\Delta}}.
\end{aligned}$$

(2b) Here, we consider the subcase $B_1 > \Delta$, which happens with probability $e^{-\Delta}$. Then we can reschedule and the cost is simply equal to what we originally would have had plus the waiting time of client 2 so far (equal to Δ):

$$\mathbb{E}[C^{3,2}_{\text{adapt}}(\Delta) \mid B_1 > \Delta] = (1-\omega)\Delta + \mathbb{E}[C^{3,2}_{\text{adapt}}(\Delta)].$$

Obviously, the cost $\mathbb{E}[C^{3,2}_{\text{adapt}}(\Delta)]$ is still unknown at this stage, but we can solve the simple equation to find them after the third and last subcase.

(2c) This last subcase occurs when $B_1 < \Delta < B_1 + B_2$, which happens with probability

$$\mathbb{P}(B_1 < \Delta < B_1 + B_2) = \int_0^{\Delta} \int_{\Delta - b_1}^{\infty} e^{-b_1} e^{-b_2} \,\mathrm{d}b_2 \,\mathrm{d}b_1 = \Delta e^{-\Delta}.$$

The cost so far is equal to $(1-\omega)$ times the conditional waiting time of client 2 so far:

$$\frac{1}{\Delta e^{-\Delta}} \int_0^{\Delta} \int_{\Delta - b_1}^{\infty} (1-\omega)\, b_1 e^{-b_1}\, e^{-b_2} \,\mathrm{d}b_2 \,\mathrm{d}b_1 = \frac{\Delta(1-\omega)}{2}.$$

Now the system has, at rescheduling epoch $t = \Delta$, reduced to the system with $n = 2$ clients with $k = 1$ client already present at the start. Therefore, the remaining cost after rescheduling is equal to $C^{2,1}_{\text{adapt}}(\Delta) = C^{2,0}_{\text{adapt}}(\Delta)$, which we already computed before in Eq. (3).

Combining the two main cases leads to the following expression for the cost:

$$\mathbb{E}[C^{3,2}_{\text{adapt}}(\Delta)] = \begin{cases} 1 + (\Delta - 3)\omega + \left(\dfrac{e^{\Delta}(\Delta - 1) + 1}{1 - e^{\Delta}} - 2\right)\dfrac{\Delta\omega}{1 - e^{\Delta}}, & \Delta < -\log\omega, \\ 1 + (\Delta - 3)\omega + \dfrac{\Delta(\log\omega - 2)}{1 - e^{\Delta}}\omega, & -\log\omega \leqslant \Delta < t_3^{\text{opt}}, \\ 1 + e^{-t_3^{\text{opt}}} + (t_3^{\text{opt}} - 2)\omega, & \Delta \geqslant t_3^{\text{opt}}, \end{cases} \tag{6}$$

with $t_3^{\text{opt}} = -\mathcal{W}_{-1}(-\omega/e)-1$. Fig. 15 shows a plot of the cost versus Δ for this model for $\omega = 0.2$. Note the discontinuities at $\Delta = 1.609$, the optimal arrival time of client 2 in the $n = 2, k = 0$ model, and at $\Delta = 2.994$, the optimal arrival time of client 3 in the $n = 3, k = 2$ model.

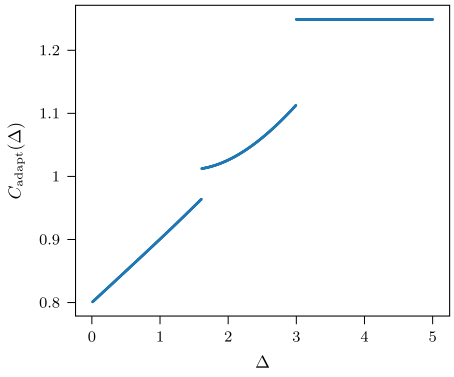


Fig. 15. Cost versus Δ for $n = 3$, $k = 2$ and $\omega = 0.2$ with homogeneous exponential service times.

Case 3: $n = 3,\ k = 0$

The final model in this appendix is the standard model with $n = 3$ clients. This model is particularly relevant because it is the smallest instance where the cost may *decrease* while increasing the rescheduling time Δ for certain intervals.

Lindley's recursion gives the following expressions:

$$\begin{aligned} I_1 &= 0, & W_1 &= 0, \\ I_2 &= \max\{t_2 - B_1, 0\}, & W_2 &= \max\{B_1 - t_2, 0\}, \\ I_3 &= \max\{t_3 - B_1 - B_2 - I_2 - W_2, 0\}, & W_3 &= \max\{B_1 + B_2 + I_2 \\ & & &\quad + W_2 - t_3, 0\}. \end{aligned}$$

The objective function is equal to

$$\begin{aligned} f(t_1, t_2, t_3 \mid k = 0) &= \omega(\mathbb{E}[I_2] + \mathbb{E}[I_3]) + (1-\omega)(\mathbb{E}[W_2] + \mathbb{E}[W_3]) \\ &= \omega(t_3 - 2) + e^{-t_2}(1-\omega) + e^{-t_3}(1 - t_2 + t_3) + e^{t_2 - t_3}. \end{aligned} \tag{7}$$

This function can only be minimized numerically, yielding the optimal arrival times $t_2 = t_2^{\text{opt}} := 1.826$ and $t_3 = t_3^{\text{opt}} := 3.699$ when $\omega = 0.2$. The corresponding cost is equal to 0.693. Instead of providing the complete analysis to find $\mathbb{E}[C^{3,0}_{\text{adapt}}(\Delta)]$, we immediately show how this cost behaves as a function of Δ by plotting them in Fig. 16. In the right panel we focus only on the cost in the interval $1.826 < \Delta < 3.699$, i.e., between the optimal arrival times of clients 2 and 3. This is probably the most surprising part of Fig. 16, in particular the part where the cost *decreases* as Δ increases. This pattern has also been observed in Experiment 1 (in particular Fig. 3) and we would like to gain more insight into this surprising behavior.

To analyze the cost in the interval $t_2^{\text{opt}} < \Delta < t_3^{\text{opt}}$, we first note that in this interval, client 2 has arrived at time $t_2 = 1.826$ and client 3 is currently scheduled at $t_3 = 3.699$. To determine the cost, we need to distinguish five different cases:

i. $B_1 < t_2^{\text{opt}}$ and $t_2^{\text{opt}} + B_2 < \Delta$. In this case, both services have finished and there is a *conditional* total idle time of $\Delta - B_1 - B_2$. There are no waiting times so far. Client 3 can be scheduled immediately and no further cost is going to occur.

ii. $t_2^{\text{opt}} < B_1 < \Delta$ and $B_1 + B_2 < \Delta$. In this case, both services have finished and there is a *conditional* total idle time of $\Delta - B_1 - B_2$. Additionally, client 2 experienced a waiting time of $B_1 - t_2^{\text{opt}}$. Client 3 can be scheduled immediately and no further cost is going to occur.

iii. $B_1 < t_2^{\text{opt}}$ and $t_2^{\text{opt}} + B_2 > \Delta$. In this case, client 2 had no waiting time and there was an idle time of $t_2^{\text{opt}} - B_1$. To determine when to optimally schedule client 3, note that the system now reduces to the $n = 2, k = 1$ system (which is equivalent to $n = 2, k = 0$ discussed earlier).

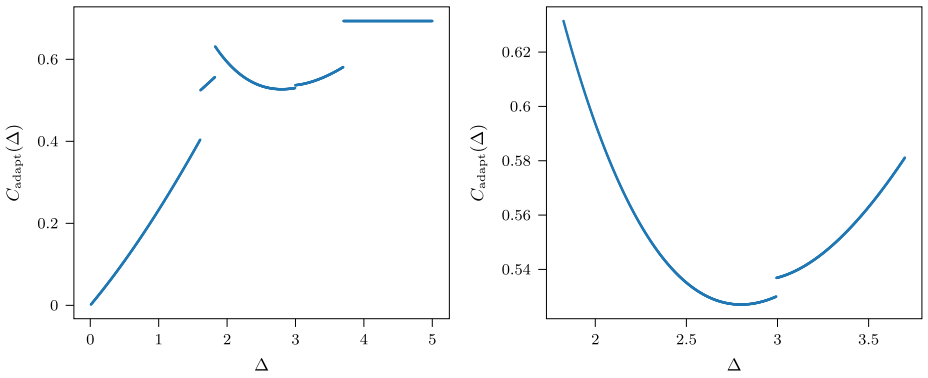


Fig. 16. Cost as a function of Δ, for $n = 3$, $k = 0$ and $\omega = 0.2$, with homogeneous exponential service times. At the right, a zoomed-in version to focus on the interval $t_2^{\text{opt}} < \Delta < t_3^{\text{opt}}$.

iv. $t_2^{\text{opt}} < B_1 < \Delta$ and $B_1 + B_2 > \Delta$. In this case, client 2 had a waiting time of $B_1 - t_2^{\text{opt}}$ and there was no idle time. As in Case *iii*, the system now reduces to the $n = 2, k = 1$ system.

v. $B_1 > \Delta$. The conditional waiting time of client 2 is equal to $\Delta - t_2^{\text{opt}}$ and the system reduces to the $n = 3, k = 2$ system discussed before. This explains the discontinuity at $\Delta = 2.994$.

So why does the cost decrease in Fig. 16 between $\Delta = 1.826$ and $\Delta = 2.8$? To understand this, we need to look at the five contributions *i*–*v* to the cost. Without going into too much detail, it can be verified that parts *iii* and *v* are decreasing, while the others are increasing. In this numerical example, the decrease of *iii* and *v* is stronger than the increase in $i + ii + iv$. Let us zoom in on Case *iii*, which happens with probability

$$\mathbb{P}(B_1 < t_2^{\text{opt}}, B_2 > \Delta - t_2^{\text{opt}}) = \int_0^{t_2^{\text{opt}}} \int_{\Delta - t_2^{\text{opt}}}^{\infty} e^{-b_1} e^{-b_2} \, \mathrm{d}b_2 \, \mathrm{d}b_1 = e^{-\Delta}\left(e^{t_2^{\text{opt}}} - 1\right).$$

The contribution of this case to the total cost in the interval $t_2^{\text{opt}} < \Delta < t_3^{\text{opt}}$ is equal to

$$\begin{aligned}
&\int_0^{t_2^{\text{opt}}} \int_{\Delta - t_2^{\text{opt}}}^{\infty} \omega(t_2^{\text{opt}} - b_1) e^{-b_1} e^{-b_2} \, \mathrm{d}b_2 \, \mathrm{d}b_1 \\
&\quad + \mathbb{P}(B_1 < t_2^{\text{opt}}, B_2 > \Delta - t_2^{\text{opt}}) \mathbb{E}[C_{\text{adapt}}^{2,0}(\Delta)] \\
&= \omega e^{-\Delta}\left(1 + e^{t_2^{\text{opt}}}(t_2^{\text{opt}} - 1)\right) + e^{-\Delta}\left(1 - e^{t_2^{\text{opt}}}\right)\omega \log \omega.
\end{aligned}$$

It is readily seen that this depends on Δ only through the factor $e^{-\Delta}$, which is clearly decreasing exponentially.

The main takeaway message, however, is that the cost function makes an upward jump whenever Δ is equal to one of the clients' arrival times. This implies that rescheduling at arrival times is, in principle, never optimal. The reason is that when scheduling at (or very soon after) a client's arrival, one loses the option to postpone the arrival if needed. In many real-life applications, obviously, the situation is more subtle, because it is not realistic (or not appreciated) to reschedule clients right before they are supposed to arrive. In these cases, a trade-off should be sought, for example by fixing appointment times within a time interval of length τ after each rescheduling epoch, as discussed in Experiment 4.

Appendix B. Pseudocode of the simulation

The following pseudocode has been used to simulate an experiment where we periodically reschedule with rescheduling time Δ.

Algorithm 2: `SimulateAdaptiveScheduling`$((\mathbb{E}B_i)_{i=1}^n, (\mathbb{S}(B_i))_{i=1}^n, \omega, \Delta)$

Result: Cost of periodically adapted schedule, with updates at times $\tau_m = m\Delta$

$(B_1, \dots, B_n) :=$ `GenerateServiceTimes`$((\mathbb{E}B_i)_{i=1}^n, (\mathbb{S}(B_i))_{i=1}^n)$
$B_{1,\text{start}} = 0$
$B_{1,\text{end}} = B_1$
$m = 0$
$(t_1, \dots, t_n) :=$ `Schedule`$((\mathbb{E}B_i)_{i=1}^n, (\mathbb{S}(B_i))_{i=1}^n, \omega \mid k = 1, u = 0)$
while $t_n > \tau_m$ **do**
  `arrivals` $= \{i \in \{1, \dots, n\} : t_i \in (\tau_m, \tau_{m+1}]\}$
  for $j \in$ `arrivals` **do**
    $B_{j,\text{start}} = \max\{B_{j-1,\text{end}}, t_j\}$
    $B_{j,\text{end}} = B_{j,\text{start}} + B_j$
  end
  $\ell :=$ max{`arrivals`} // `last arrival`
  if $B_{\ell,\text{end}} < \tau_{m+1}$ **then** // `system is idle`
    $(t_{\ell+1}, \dots, t_n) =$ `Schedule`$((\mathbb{E}B_i)_{i=\ell+1}^n, (\mathbb{S}(B_i))_{i=\ell+1}^n, \omega \mid k = 1, u = 0)$
  else // `a client is in service`
    `waiting_clients` $= \{i \in \{1, \dots, \ell\} : B_{i,\text{start}} > \tau_{m+1}\}$
    if `waiting_clients` $= \emptyset$ **then**
      $c := \ell$ // `client in service`
    else
      $c :=$ min{`waiting_clients`} $- 1$
    end
    $k :=$ |`waiting_clients`| $+ 1$
    $u := \tau_{m+1} - B_{\text{start},c}$
    $(t_{\ell+1}, \dots, t_n) =$ `Schedule`$((\mathbb{E}B_i)_{i=\ell+1}^n, (\mathbb{S}(B_i))_{i=\ell+1}^n, \omega \mid k, u)$
  end
  $m := m + 1$
end
// `Lindley recursion`
$W_1 = I_1 = 0$
for $i = 2$ **to** n **do**
  $L_i := (t_{i-1} + W_{i-1} + B_{i-1}) - t_i$
  $W_i = \max\{L_i, 0\}$
  $I_i = \max\{-L_i, 0\}$
end
return $\omega \sum_{i=1}^n I_i + (1 - \omega) \sum_{i=1}^n W_i$

References

Ahmadi-Javid, A., Jalali, Z., Klassen, K., 2017. Outpatient appointment systems in healthcare: a review of optimization studies. European J. Oper. Res. 258, 3–34.

Asmussen, S., 2003. Applied Probability and Queues, second ed. Springer, New York.

Berg, B., Denton, B., Erdoğan, S., Rohleder, T., Huschka, T., 2014. Optimal booking and scheduling in outpatient procedure centers. Comput. Oper. Res. 50, 24–37.

Çayırlı, T., Veral, E., Rosen, H., 2006. Designing appointment scheduling systems for ambulatory care services. Health Care Manage. Sci. 9, 47–58.

Chen, R., Robinson, L., 2014. Sequencing and scheduling appointments with potential call-in patients. Prod. Oper. Manage. 23, 1522–1538.

de Kemp, M., Mandjes, M., Olver, N., 2021. Performance of the smallest-variance-first rule in appointment sequencing. Oper. Res. 69, 1651–1959.

Decerle, J., Grunder, O., El Hassani, A., Barakat, O., 2018. A memetic algorithm for a home health care routing and scheduling problem. Oper. Res. Health Care 16, 59–71.

Doğru, A., Melouk, S., 2019. Adaptive appointment scheduling for patient-centered medical homes. Omega 85, 166–181.

Erdoğan, S., Denton, B., 2013. Dynamic appointment scheduling of a stochastic server with uncertain demand. INFORMS J. Comput. 25, 116–132.

Erdoğan, S., Gose, A., Denton, B., 2015. Online appointment sequencing and scheduling. IIE Trans. 47, 1267–1286.

Ghosh, M., Kuiper, A., Mahes, R., Maragno, D., 2023. Learn global and optimize local: a data-driven methodology for last-mile routing. Comput. Oper. Res. 159, 106312.

Kong, Q., Lee, C., Teo, C., Zheng, Z., 2016. Appointment sequencing: Why the smallest-variance-first rule may not be optimal. European J. Oper. Res. 255, 809–821.

Kuiper, A., 2016. Optimal Appointment Scheduling in Healthcare (Ph.D. thesis). University of Amsterdam.

Kuiper, A., Kemper, B., Mandjes, M., 2015. A computational approach to optimized appointment scheduling. Queueing Syst. 79, 5–36.

Kuiper, A., Lee, R., 2022. Appointment scheduling for multiple servers. Manage. Sci. 68, 7422–7440.

Kuiper, A., Mandjes, M., de Mast, J., Brokkelkamp, R., 2022. A flexible and optimal approach for appointment scheduling in healthcare. Decis. Sci. J. 54, 85–100.

Lindley, D., 1952. The theory of queues with a single server. Math. Proc. Camb. Phil. Soc. 48, 277–289.

Liu, R., Yuan, B., Jiang, Z., 2019. A branch-and-price algorithm for the home-caregiver scheduling and routing problem with stochastic travel and service times. Flex. Serv. Manuf. J. 31, 989–1011.

Mahes, R., Mandjes, M., Boon, M., Taylor, P., 2023. Adaptive scheduling in service systems: a dynamic programming approach. European J. Oper. Res. 312, 605–626.

Mak, H., Rong, Y., Zhang, J., 2014. Sequencing appointments for service systems using inventory approximations. Manuf. Serv. Oper. Manage. 16, 251–262.

Mak, H., Rong, Y., Zhang, J., 2015. Appointment scheduling with limited distributional information. Manage. Sci. 61, 316–334.

Neuts, M., 1994. Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach. Courier Corporation.

Pegden, C., Rosenshine, M., 1990. Scheduling arrivals to queues. Comput. Oper. Res. 17, 343–348.

Robinson, L., Chen, R., 2003. Scheduling doctors' appointments: optimal and empirically-based heuristic policies. IIE Trans. 35, 295–307.

Spliet, R., Desaulniers, G., 2015. The discrete time window assignment vehicle routing problem. European J. Oper. Res. 244, 379–391.

Spliet, R., Gabor, A., 2014. The time window assignment vehicle routing problem. Transp. Sci. 49, 721–731.

Stein, W., Côté, M., 1994. Scheduling arrivals to a queue. Comput. Oper. Res. 21, 607–614.

Tijms, H., 1994. Stochastic Models: An Algorithmic Approach. Wiley, New York.

Wang, P., 1997. Optimally scheduling N customer arrival times for a single-server system. Comput. Oper. Res. 24, 703–716.

Wang, J., Fung, R., 2014. Adaptive dynamic programming algorithms for sequential appointment scheduling with patient preferences. Artif. Intell. Med. 63, 33–40.

Wang, J., Fung, R., 2015. Dynamic appointment scheduling with patient preferences and choices. Ind. Manage. Data Syst. 115, 700–717.

Wang, W., Gupta, D., 2011. Adaptive appointment systems with patient preferences. Manuf. Serv. Oper. Manage. 13, 373–389.

Zacharias, C., Yunes, T., 2020. Multimodularity in the stochastic appointment scheduling problem with discrete arrival epochs. Manage. Sci. 66, 744–763.

Zhan, Y., Wang, Z., Wan, G., 2021. Home service routing and appointment scheduling with stochastic service times. European J. Oper. Res. 288, 98–110.