

# Do Today's Work Today: Don't Send Patients Away!

An appointment scheduling algorithm that can deal with walk-in



*Master Thesis Industrial Engineering and Management*

University of Twente, Enschede & Academic Medical Centre, Amsterdam

**Joost Veldwijk**

*August 2012*

Academic Medical Centre (AMC)  
Meibergdreef 9  
1105 AZ, Amsterdam



University of Twente  
Drienerlolaan 5  
7522 NB, Enschede

**UNIVERSITY OF TWENTE.**

Do Today's Work Today: Don't Send Patients Away!  
An appointment scheduling algorithm that can deal with walk-in

**Author:**

J.S. Veldwijk, University of Twente  
Industrial Engineering and Management  
Studentnumber: s0182826

**Graduation committee:**

dr. ir. I.M.H. Vliegen, University of Twente  
School of Management and Governance  
Department: Industrial Engineering and Business Information Systems (IEBIS)

dr. N. Litvak, University of Twente  
Faculty of Electrical Engineering, Mathematics and Computer Science  
Department: Stochastic Operations Research (SOR)

ir. A. Braaksma, AMC & University of Twente  
Department: Quality and Process Innovation (KPI) (AMC)  
Faculty of Electrical Engineering, Mathematics and Computer Science (University of Twente)  
Department: Stochastic Operations Research (SOR)

# Summary

Traditionally diagnostic facilities schedule appointments for all patients that require an examination. Allowing patients to walk in without an appointment reduces access times. It also creates the possibility to combine outpatient consultations and diagnostic examinations on one day, which speeds up the diagnostic process. Since not all patients can walk in, our goal is to develop an algorithm that generates appointment schedules with which both patients with an appointment and walk-in patients can be served. The generated schedule should prescribe the number of appointments to schedule per day and the moment on the day to schedule these appointments. We maximize the fraction of walk-in patients that can be served on the day of their arrival, while satisfying an access time service level norm for patients with an appointment. We conducted our research at the Academic Medical Centre (AMC), a large academic hospital in Amsterdam.

A methodology developed in earlier research [37] generates good schedules by complete enumeration. However, schedules of realistic size cannot be generated since evaluating all solutions is too time consuming. We build on this earlier research, but to achieve computational efficiency we use heuristics that are based on workload levelling. From our literature review it appeared that local search techniques are often useful to improve the schedules found with heuristics, so we also use local search techniques in our algorithm. To generate an appointment schedule, we first determine a capacity cycle that indicates how many appointments should be scheduled per day in a planning cycle (i.e. one week). The next step is to allocate these appointments over the available time slots of a day.

For both capacity cycle generation and day schedule generation we tested several local search techniques to obtain improved appointment schedules. These tests were performed on small instances, such that we could compare the outcomes of our heuristics to the solutions found with complete enumeration. We first tested our capacity cycle generating heuristics, while using complete enumeration for making day schedules. Secondly we tested all combinations of capacity cycle heuristics, day schedule heuristics and local search techniques. We tested for 36 problem instances. The best performing heuristic with respect to the fraction of walk-in patients served on the day of their arrival, determines the number of appointments to schedule per day based on the expected number of arriving walk-in patients per day. Allocating appointments to time slots of a day gives the best performance when we generate an initial schedule based on a heuristic that allocates appointments to time slots with few expected walk-in arrivals, in combination with a simple random search technique. This confirms findings from literature, that state that random search outperforms more advanced local search methods for certain cases.

The best performing combination of heuristics deviates less than 0.5% from complete enumeration on average, defers in the worst case 4.99% more walk-in patients than complete enumeration and finds the same solution as complete enumeration in 72% of the test instances. Deviations from complete enumeration were in most cases caused by a tight allowed access time. A tight access time makes it harder to allocate appointments over the days in a cycle, which makes the schedule less flexible. This best performing combination of heuristics needs less than 5 minutes to generate a (small) appointment schedule on average, while complete enumeration needs more than 8 hours on average. However, the

performance differences among the best performing combinations of heuristics were small, so we decided to test the three best performing combinations in our case study.

In the case study we used data of the CT-scan facility of the AMC, gathered in 2008. In our first tests on this large instance, it appeared that the approach we used to evaluate day schedules (as described in [37]) gave unreliable results. Therefore we adapted this approach, which finally led to reliable values for the performance of day schedules. As a benchmark we use day schedule generation rules from literature. Because methods to generate capacity cycles are scarce in literature, we decided to use our capacity cycle heuristics to generate capacity cycles for the benchmarks. We observe from the case study that all combinations of heuristics we tested perform significantly better with respect to the fraction of walk-in patients served on the day of their arrival than the benchmarks. Our algorithm deferred 75.5% less patients than the best performing benchmark and 99.43% of the walk-in patients could be served on the day of their arrival (with a workload of 62.3%). Furthermore, our algorithm was able to find an appointment schedule for the case study instance within 1.5 hours runtime. When evaluating the resulting appointment schedules, we see that appointments are planned in quiet periods of the day with respect to arriving walk-in patients. This is according to our expectations, since in this way workload gets balanced over the day. We also observe that the waiting behaviour of walk-in patients is exploited. All resources are used for appointments at the start of the day, in such way that arriving walk-in patients in the first time slots can be served after this early block of appointments. In a test on a scaled-up case study instance with a workload of 85%, this algorithm serves 94.16% of the walk-in patients on the day of their arrival and it defers 55.8% less patients than the best performing benchmark. The corresponding appointment schedule was generated in 3.2 hours. These results are promising, however more extensive testing is necessary to confirm these findings.

Before our algorithm can be implemented in practice, we recommend to perform a simulation study to test the influence of the assumptions we model. Simulation is a flexible method, so characteristics from practice that we do not model can then be incorporated. One can think of different patient types, stochastic service times and stochastic waiting times for walk-in patients. We advise to execute this research in close collaboration with the CT-scan facility, so that commitment and trust in the planning algorithm can be created. It is also important to optimise the parameters of the local search techniques used in our algorithm. The choices we make show promising results, however it might be that performance with respect to runtime or the fraction of walk-in patients served on the day of their arrival can be improved. Since deviations from complete enumeration are mainly caused by our capacity cycle generating heuristics, we advise to conduct further research to better capacity cycle generating heuristics. A more advanced local search technique than we use, based on random search for example, might be a good option.

We can conclude that the algorithm we develop in this thesis fills a gap in appointment scheduling literature. Our algorithm is able to generate appointment schedules of realistic size with good performance in reasonable time. Furthermore, our algorithm brings successful implementation of one-stop-shop in healthcare (and other businesses that use appointment systems) a step closer.

# Contents

<b>List of Symbols</b>	<b>vi</b>
<b>List of Abbreviations</b>	<b>vii</b>
<b>Preface</b>	<b>viii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Academic Medical Centre Amsterdam . . . . .	1
1.2 Problem description and scope . . . . .	1
1.3 Research objectives . . . . .	3
1.4 Report overview . . . . .	4
<b>2 Problem analysis</b>	<b>5</b>
2.1 Desired situation . . . . .	5
2.2 Approach of Kortbeek et al. [37] . . . . .	6
2.3 Models and algorithm . . . . .	7
2.4 Summary & conclusion . . . . .	9
<b>3 Literature review</b>	<b>10</b>
3.1 Appointment scheduling in outpatient clinics . . . . .	10
3.2 Finding good capacity cycles . . . . .	11
3.3 Finding good day schedules . . . . .	12
3.4 Summary & conclusion . . . . .	15
<b>4 Algorithm development</b>	<b>16</b>
4.1 Algorithm structure . . . . .	16
4.2 Heuristics for generating capacity cycles . . . . .	17
4.3 Heuristics for generating day schedules . . . . .	21
4.4 Local search for appointment schedule improvement . . . . .	24
4.5 Summary & conclusion . . . . .	27
<b>5 Algorithm testing</b>	<b>28</b>
5.1 Approach . . . . .	28
5.2 Test settings . . . . .	29
5.3 Results . . . . .	33
5.4 Summary & conclusion . . . . .	38

<b>6 Case study</b>	<b>39</b>
6.1 Making a CT-scan . . . . .	39
6.2 Case study setup . . . . .	39
6.3 Results . . . . .	43
6.4 Summary & conclusion . . . . .	46
<b>7 Conclusions and recommendations</b>	<b>47</b>
7.1 Conclusions . . . . .	47
7.2 Discussion . . . . .	48
7.3 Suggestions for further research . . . . .	49
<b>Bibliography</b>	<b>53</b>
<b>Appendices:</b>	
<b>A Day schedule evaluation example</b>	<b>I</b>
<b>B Arrival rates test setting</b>	<b>II</b>
<b>C Results of capacity cycle generation tests</b>	<b>IV</b>
<b>D Results of day schedule generation tests</b>	<b>VIII</b>
<b>E Numerical results in the case study</b>	<b>XXI</b>
<b>F Case study test under 85% workload</b>	<b>XXII</b>

# List of Symbols

$D$	Length of the planning cycle
$T$	Total number of time slots per day
$d$	Day index ( $d = 1, \dots, D$ )
$t$	Time slot index ( $t = 1, \dots, T$ )
$h$	Length of a time slot
$g$	Maximum number of time slots a walk-in patient is willing to wait for service
$R$	Number of available resources
$k^d$	Number of time slots reserved for appointments on day $d$
$K$	Capacity cycle ( $k^1, \dots, k^D$ )
$\Gamma$	Total number of places reserved for appointments over the planning cycle
$C^d$	Appointment schedule on day $d$ , $C^d = (c_1^d, \dots, c_T^d)$
$c_t^d$	Maximum number of appointments to schedule in time slot $t$ on day $d$
$A^d$	Number of arriving walk-in patients on day $d$
$\gamma^d$	Appointment request rate on day $d$
$\lambda^d$	Initial appointment request rate on day $d$
$\chi_t^d$	Walk-in arrival rate on day $d$ and time slot $t$

# List of Abbreviations

AMC	Academic Medical Centre
CAS	Cyclic Appointment Schedule
CT	Computed Tomography
FCFS	First Come First Served
GA	Genetic Algorithm
IBFI	Individual Block, Fixed Interval
KPI	Kwaliteit en Proces Innovatie (Quality and Process Innovation)
LS-CC	Local Search - Capacity Cycles
LS-GA	Local Search - Genetic Algorithm
LS-KK	Local Search - Kaandorp & Koole
LS-RS	Local Search - Random Search
MDP	Markov Decision Process
MRI	Magnetic Resonance Imaging
ORAHS	Operational Research Applied to Health Services
SA	Simulated Annealing



# Preface

This report is the final element of my study to become an MSc. in Industrial Engineering and Management. Before I got a graduation assignment I was not sure whether its emphasis should be on solving a practical problem or doing research, since I am interested in both. The assignment I executed at the Academic Medical Centre (AMC) in Amsterdam was an excellent opportunity to find out whether doing research suits me and the outcomes of my research can hopefully be used in practice in the near future. Therefore was working on this assignment a very nice experience for me. In the process of doing my research I got help from several people, whom I like to thank.

First I thank Ingrid Vliegen and Nelly Litvak, my supervisors from the University of Twente. Ingrid offered me the opportunity to execute my graduation project at a hospital in Amsterdam, which had my preference. Besides that she always had time to answer my questions and she gave me the opportunity to present the content of my thesis at the Operational Research Applied to Health Services (ORAHS) conference in Enschede. Nelly helped me a lot with explaining some mathematical issues. Without her help I would not have been able to present my thesis here in August. I would also like to thank all my colleagues of the KPI department at the AMC. In the discussions we had during lunches I got a good understanding of the challenges in improving quality and processes in a hospital, from many different points of view. Special thanks go to Aleida Braaksma, my external supervisor at the AMC. In our weekly discussions she always let me think about other possible solutions for problems I encountered and her extensive reviews of my work helped me to create a report to be proud of. I also thank Nikky Kortbeek and Maartje Zonderland who made time for me to answer my questions about their model.

Besides the persons I worked with during my graduation project, I thank my parents. They have always supported me and were interested in my progress and findings. Last but not least I thank my girlfriend Annika. She helped me to stay motivated and was always there for me.

Amsterdam, August 2012

Joost Veldwijk

# Chapter 1

## Introduction

Rising healthcare costs and increasing patient expectations about the quality of care emphasize the need for process innovations in healthcare. Timely access and short waiting times are important characteristics of a high quality care provider. However, many consultations and examinations in hospitals still take place on appointment. This delays the treatment process of a patient, which can deteriorate the quality of care. In case appointment schedules would take both patients with an appointment and walk-in patients into consideration, the diagnostic process speeds up and patient satisfaction increases. Therefore we develop an appointment scheduling algorithm, based on earlier research [37], that can deal with both walk-in patients and patients with an appointment.

In Section 1.1 we start with an introduction of the Academic Medical Centre (AMC), which is the hospital where we conduct our research. We describe the problem as faced by the AMC in Section 1.2 and present our research objectives in Section 1.3. In Section 1.4 we give an overview of the remainder of this thesis.

### 1.1 Academic Medical Centre Amsterdam

We conduct our research at the AMC, a large academic hospital in Amsterdam that is connected to the University of Amsterdam. The AMC is the preferred hospital for about 200,000 residents living close to it. Because of the academic nature of the AMC, the hospital serves patients with more complicated diseases. These patients come from all over the Netherlands.

The AMC has three main tasks: offering education to medicine students, giving patients the care they need, and performing high level medical scientific research. This thesis is written in collaboration with the department Quality and Process Innovation (Kwaliteit en Proces Innovatie (KPI)). This department contributes to the three main tasks of the AMC by supporting other departments in improving their quality of care. KPI develops models and methods that can be applied within the AMC to obtain these structural quality improvements, for example in the fields of patient logistics and evidence based medicine.

### 1.2 Problem description and scope

Diagnostic examinations often occur in combination with outpatient consultations. However, diagnostic facilities in the AMC traditionally work with appointment systems. This means that patients who need a diagnostic examination have to make an appointment after their outpatient consultation, which may cause high access times. We define access time as the time (in days) between the day on which the patient makes an appointment and the day the patient is served.

Since fast diagnosis is important for the quality of care, it would be preferable to have no access time at all [41]. Having no access time, the one-stop-shop idea [5], is also preferable from a patient's point of view. Without access time, patients only have to visit the hospital once for an outpatient consultation and diagnostic examination. This saves time for the patient and treatment can be started earlier.

Allowing patients to walk in at diagnostic facilities without an appointment would reduce access times. The possibility to combine outpatient consultations and diagnostic examinations on one day, which speeds up the diagnostic process, would then also arise. However, diagnostic examinations cannot be planned in advance, because the decision to perform a diagnostic examination usually depends on the findings of the doctor during the consultation. This gives reason to leave some space in appointment schedules of diagnostic facilities to accommodate for these walk-in patients. Successful implementations of systems with walk-in report to improve patient satisfaction and resource utilisation while reducing healthcare costs [42, 45].

Allowing walk-in patients to enter diagnostic examinations would still result in a consultation followed by a diagnostic examination, but then in most cases on the same day. Diagnostic facilities with low-variable and short service times have the highest chance to successfully adopt such a strategy. A highly variable service time causes a higher risk on large waiting times [51] and that is what we want to avoid from a patient point of view.

Patrick [43] argues that allowing access exclusively for walk-in patients is not a good idea with respect to throughput and costs. Since demand is then highly uncertain, workload is often unbalanced over the day which results in under- and over- utilisation of resources. Earlier research at the AMC shows that a combination of scheduled appointments and unscheduled appointments (walk-in) has more advantages than a setting with only walk-in patients [38]. This is mainly a practical issue, since some patients prefer an appointment and some examinations need the presence of specialists who are not always available. To achieve this combination of scheduled appointments and walk-in patients in outpatient clinics, Kortbeek et al. [37] developed a method to design appointment schedules. Their algorithm is able to generate small appointment schedules (e.g. 8 time slots per day) that can deal with walk-in. However, the computational complexity of their algorithm is very high such that appointment schedules for practice cannot be generated in reasonable time. Because their model is not able to generate appointment schedules of realistic size, its performance and applicability in practice is uncertain. Because we are looking for a method that can be used in practice, we have to adapt the model of Kortbeek et al. [37] to overcome these practical issues. This leads us to the following problem statement:

*"The potential of combining scheduled and unscheduled patient arrivals for diagnostic facilities is clear. However, a method to develop appointment schedules for diagnostic examinations in practice is lacking."*

We position our research in the healthcare planning and control framework as developed by Hans et al. [25]. This framework spans four hierarchical levels of control and four managerial areas as displayed in Figure 1.1. Since our research aims to design appointment schedules, we position it as resource capacity planning on a tactical level. It is positioned as resource capacity planning because appointment scheduling is a typical activity in this managerial area [25]. Our goal is to make appointment schedules that can be used for some time without allocating actual patients to the schedule. Therefore our algorithm can be used on a tactical level. The schedules generated by our algorithm can be used in operational planning, where actual patients are allocated to the schedule. On the other hand, the schedules that our algorithm generates are based on strategic decisions such as the number of resources a facility needs.

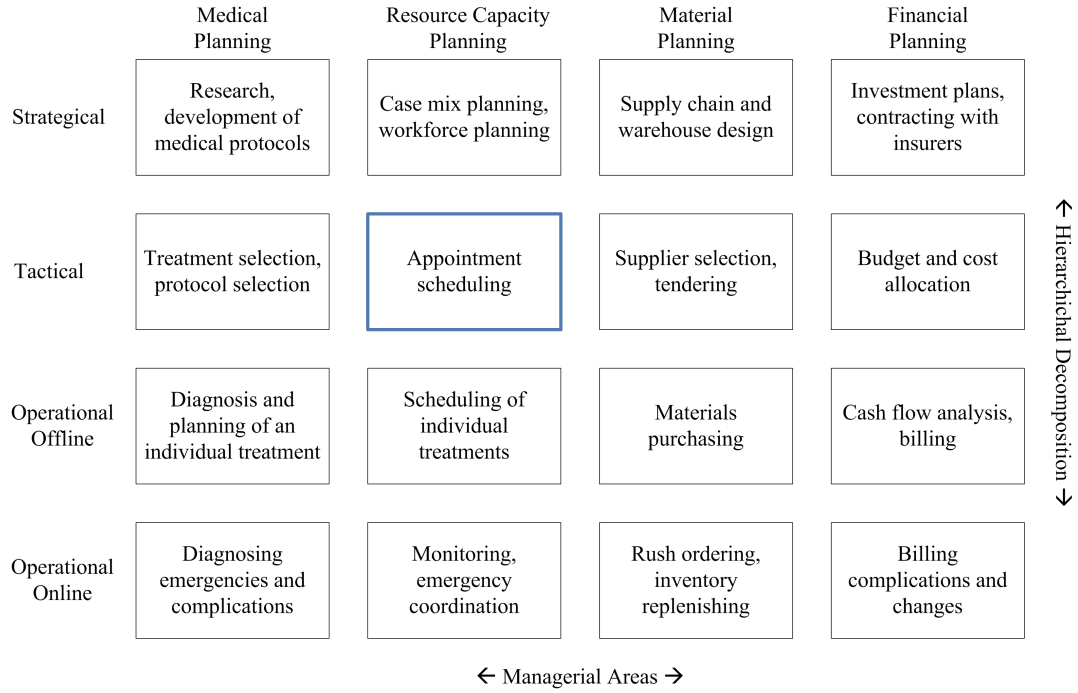


Figure 1.1: Healthcare planning and control framework

### 1.3 Research objectives

Kortbeek et al. [37] developed an algorithm that can generate appointment schedules, but the computational complexity for calculating these appointment schedules is high. They indicate that a challenge lies in achieving numerical efficiency of their algorithm. The purpose of our research is therefore to adapt the model of Kortbeek et al. [37] such that appointment schedules of realistic size can be generated. Because our algorithm should be used in practice, it is important that our algorithm is relatively fast (i.e. a planner has limited time to generate an appointment schedule).

As said, we use the model and algorithm as presented by Kortbeek et al. [37] as the starting point of our research. Their algorithm generates cyclic appointment schedules. This means that the planning horizon in their model consists of several days. For all of these days an appointment schedule has to be made, that combines scheduled appointments and walk-in arrivals. In Kortbeek et al. [37] a *good* appointment schedule has two main characteristics. The first characteristic is that access time for patients who require an appointment is lower than an access time norm set by the management of the facility. Access time is dependent on the number of appointments allocated to the days over the planning horizon. The second characteristic is that the percentage of walk-in patients served on the day of their arrival is maximised. This characteristic is dependent on the way appointments are scheduled over the day. Since our algorithm is based on that of Kortbeek et al. [37], our appointment schedules should also possess these two characteristics, however our algorithm should generate appointment schedules of realistic size (i.e. more than 30 time slots per day). We identified the following research objectives to achieve this overall research aim:

1. Find a good way to determine how many appointments should be planned per day;
2. Find a good way to allocate appointments to time slots;
3. Assess how the algorithm performs in generating feasible appointment schedules that allow for both scheduled and unscheduled appointments for the CT-scan facility of the AMC.

When we want to allocate appointments to time slots, we first have to know how many appointments we have to schedule at each day over the planning horizon. Our second research goal is to find a good and fast way of allocating appointments to time slots. Our last goal is to test the performance of our algorithm with data from the CT-scan facility of the AMC. When we answer our three research questions, we are able to deliver an appointment scheduling algorithm that can generate good appointment schedules of realistic size in little time.

## 1.4 Report overview

In this section we give an overview of the content of this report. In Chapter 2 we describe the problem as faced by the AMC. We discuss the desired output of our scheduling algorithm and we describe the algorithm of Kortbeek et al. [37]. In Chapter 3 we present an overview of literature that deals with appointment scheduling. Our main interest is in literature that combines scheduled appointments and walk-in patients in one appointment schedule. In Chapter 4 we develop heuristics that can generate appointment schedules and we discuss local search techniques that can improve the schedules found with our heuristics. In Chapter 5 we show the results of tests we executed on theoretical problem instances. In Chapter 6 we perform a case study with data of the CT-scan facility at the AMC. In Chapter 7 we end this thesis with conclusions and recommendations. In that chapter we also discuss the limitations of our study and give suggestions for further research. For clarity we present a list of symbols on page vi and a list of abbreviations on page vii.

# Chapter 2

## Problem analysis

In this chapter we give an overview of the algorithm of Kortbeek et al. [37], which we use as the starting point of our study. Section 2.1 presents the desired situation with respect to the output of the appointment scheduling algorithm we develop. In Section 2.2 we present the approach of Kortbeek et al. [37] and introduce their notation and assumptions. In Section 2.3 we describe how the algorithm of Kortbeek et al. [37] generates appointment schedules. In Section 2.4 we present a summary and our conclusions.

### 2.1 Desired situation

In the ideal situation, hospitals are able to combine walk-in patients and scheduled appointments in appointment schedules that give an optimal solution. In an optimal solution, all walk-in patients are served on the day of their arrival and all patients with an appointment request are served within the access time norm. Since an optimal solution is very hard to obtain because of computational complexity, we aim to outperform existing scheduling techniques with our algorithm. This implies that we have to make a clever choice in allocating time slots to appointments and leaving time slots free to serve walk-in patients. In Figure 2.1 we present a possible output in the desired situation, that can be used by planners to schedule appointments. Coloured blocks indicate a scheduled appointment (or the opportunity to schedule an appointment) and the white blocks indicate time slots that are left free to serve walk-in patients.

For certain diagnostic examinations, such as CT scans at the CT-scan facility, service times are relatively short and usually time slots of equal size are used for all examinations [23]. This implies that our algorithm should be able to produce good schedules consisting of many time slots. The target number of time slots that should be included is 34 per day: a resource at the CT-scan facility of the AMC is used for 8.5 hours per day and the average service time is 15 minutes. The same applies for the used planning cycle, this should preferably be a maximum of approximately 10 working days. Our algorithm should also be able to make a good appointment schedule in case multiple resources are available. For the CT-scan facility of the AMC this would mean that a schedule should be generated for a maximum of three resources.

Kortbeek et al. [37] have developed an algorithm that is able to generate good appointment schedules. However, the runtime to generate a schedule for a small instance (smaller than the desired parameter settings as described, more on this in Section 2.2) is about nine hours on an Intel 3.2 Ghz PC with 4Gb of RAM. Therefore, the method of Kortbeek et al. [37] is not directly applicable in practice. This means that our algorithm should generate good appointment schedules under the same assumptions as in Kortbeek et al. [37], but much faster. A trade-off should be made between runtime

	8:00 – 8:15	8:15 – 8:30	8:30 – 8:45	8:45 – 9:00	...
Monday	Reserved for appointment	Reserved for appointment	Reserved for appointment	Reserved for appointment	
Tuesday		Reserved for appointment			
Wednesday	Reserved for appointment	Reserved for appointment		Reserved for appointment	
Thursday	Reserved for appointment	Reserved for appointment			
Friday					
Monday	Reserved for appointment	Reserved for appointment	Reserved for appointment	Reserved for appointment	
Tuesday		Reserved for appointment			
Wednesday	Reserved for appointment	Reserved for appointment		Reserved for appointment	
...					

Figure 2.1: Visual representation of ideal output of our algorithm

and schedule performance. Preferably a schedule can be generated in little time. However, a characteristic of tactical schedules is that they only have to be updated once in a few months (e.g. seasonal updates). We expect that a better performance can be obtained when the runtime is higher (more possible schedules can be evaluated), but a planner cannot wait too long to receive a new appointment schedule (e.g. claim on hardware, possibility to make changes in the schedule). From a practical point of view, we think that the maximum runtime of our algorithm should not exceed one day.

## 2.2 Approach of Kortbeek et al. [37]

Kortbeek et al. [37] present a method to design cyclic appointment schedules that can be used at diagnostic facilities that serve patients with an appointment and walk-in patients. Their method consists of two models: one to evaluate the access process of patients with an appointment (Access model) and one to evaluate the day process of appointments and walk-in patients (Day model). The Access model evaluates the time a patient has to wait from the day an appointment is made until the day of service. The Day model evaluates the time a walk-in patient has to wait from the time of arrival until service. It may happen that the facility is temporary congested, which results in high waiting times for walk-in patients. If a walk-in patient has to wait more than  $g$  time slots to receive service, the patient is offered an appointment on another day. Kortbeek et al. [37] refer to such patients as *deferred* patients. We discuss the Access- and Day model in more detail in Section 2.3. The goal of Kortbeek et al. [37] is to minimise the number of walk-in patients who have to be deferred. This is equal to maximising the fraction of walk-in patients who can be served on the day they arrive. This should be done under the constraint that the access time of patients with an appointment request is lower than a preset access time service level norm. The idea is that the management of the hospital can decide on this access time service level norm (e.g. 95% of the patients requesting an appointment should be served within 10 days).

**Assumptions.** Because walk-in demand and appointment demand are often cyclic [3, 16], Kortbeek et al. [37] propose a Cyclic Appointment Schedule (CAS) with a length of  $D$  days,  $R$  resources and  $T$  time slots of length  $h$  on each day.<sup>1</sup> This CAS is represented by  $C = (C^1, \dots, C^D)$ , where  $C^d$  indicates the appointment schedule on day  $d$ . Two types of patients have to be served: patients with a scheduled

<sup>1</sup>In their numerical example Kortbeek et al. [37] study a one-resource situation, with a cycle length of 5 days consisting of 8 equally sized time slots.

appointment and patients who walk in (unscheduled patients). Walk-in patients are willing to wait for treatment a maximum of  $g$  time slots after arrival. In case service cannot start within this interval, the walk-in patient is offered an appointment at a later day and the patient becomes a deferred patient. All patients with an appointment, so patients with a scheduled appointment and deferred walk-in patients, are scheduled First Come First Served (FCFS). Kortbeek et al. [37] assume a non-stationary Poisson process for the arrival of appointment requests, with initial arrival rates  $\lambda^1, \dots, \lambda^D$  for each day in the planning cycle. Patients with an appointment have priority over walk-in patients and may not show up. If a patient shows up it is assumed that he is on time, which is a reasonable reflection of reality [28]. For walk-in patients the arrival rate depends on the day  $d$  and time slot  $t$ . The arrival process of these patients is also modelled as a non-stationary Poisson process with arrival rates  $\chi_t^d$ . Service times are assumed to be the same for walk-in patients and patients with an appointment. This service time is assumed to be deterministic and equal to the length of one time slot (i.e. in the algorithm of Kortbeek et al. [37] patients always need one time slot for service). Note that a list of symbols can be found on page .

## 2.3 Models and algorithm

The performance of the Access model and the Day model is measured on different time scales (days for the Access model versus minutes for the Day model), which makes a comparison of both measures difficult. Therefore, Kortbeek et al. [37] decompose the planning process in these two models to determine performance measures for the access time of a patient with an appointment request (Access model) and waiting time for walk-in patients (Day model). Kortbeek et al. [37] link the Access model and the Day model with an iterative algorithm, to balance the scheduled and unscheduled arrivals. This iterative algorithm determines the optimal size of the group of deferred patients by gradually increasing its size during each iteration [37]. Now we will describe the Access model, the Day model and the iterative algorithm as presented by Kortbeek et al. [37] in more detail.

**The Access Model** determines how many time slots should be reserved for appointments over the planning cycle, under the constraint that this number of appointments is sufficient to meet the access time service level norm. Furthermore, the Access model determines how these appointments are divided over the days in the planning cycle. The output of the Access model is called a *capacity cycle* and is represented by  $K = (k^1, \dots, k^D)$ , where  $k^d$  represents the number of time slots that has to be reserved for appointments on day  $d$ . A capacity cycle  $K$  only indicates how many time slots have to be reserved for appointments on each day, it does not indicate *which* time slots have to be reserved for appointments, this is determined by the Day model.

**The Day model** uses the capacity cycles generated by the Access model to generate all possible *day schedules*. A day schedule indicates in which time slots appointments can be scheduled and where space for walk-in patients should be reserved. From all possible day schedules, the Day model gives the best performing day schedule per day of the capacity cycle as output. The best schedule is the one that minimises the expected number of deferred patients. A day schedule is represented by  $C^d = (c_1^d, \dots, c_T^d)$ , where  $c_t^d$  is the *maximum* number of patients that may be scheduled in time slot  $t$  on day  $d$ .<sup>2</sup> The values of  $k^d$  are thus an upper bound to the number of scheduled appointments on a certain day. We describe the steps to evaluate a day schedule in Appendix A. The combination of a capacity cycle  $K$  and its corresponding best day schedules  $C = (C^1, \dots, C^D)$  forms the CAS. This CAS represents how many and which time slots have to be reserved for appointments on each day.

---

<sup>2</sup>We explicitly use *maximum* here, because the appointment slots as placed in the best schedule found are not necessarily fully occupied by appointments. Appointments can only be scheduled in those time slots, but the actual occupation is dependent on the number of appointment requests (which may be lower than the number of reserved time slots).



**The iterative algorithm** minimises the number of deferred walk-in patients while satisfying the access time service level norm. In the first iteration of the algorithm of Kortbeek et al. [37], the expected number of deferred patients is set to zero. First, all feasible capacity cycles are determined with the Access model. Second, for each capacity cycle generated with the Access model, the Day model determines all possible schedules for each day in the cycle. For each day in each capacity cycle, the Day model selects the best day schedule generated. The last step of the Day model is to choose the combination of a capacity cycle and its corresponding set of day schedules that minimises the total number of deferred patients. If the number of deferred patients in an iteration is (much) larger than in a previous iteration, the number of time slots reserved for appointments was apparently not sufficient and more time slots should be reserved for appointments. To overcome this problem, a new iteration is started. At the start of a new iteration, the appointment request rate is updated for each day: it consists of the number of deferred patients of that day in the previous iteration added to the initial arrival rate of appointment requests and is represented by  $\gamma^d$ . This new appointment request rate is used to evaluate the Access and Day model again. This is done until there is no significant change in the number of deferred patients in two subsequent iterations anymore (i.e. the number of time slots reserved is sufficient to service all scheduled patients and all deferred patients that receive an appointment, while the remaining walk-in patients are served on the day of arrival). This balance should hold for all days in the capacity cycle, so balance in the sum of deferred patients over all days is not sufficient. Since all feasible capacity cycles for a certain  $K$  and all possible day schedules are evaluated, the algorithm of Kortbeek et al. [37] returns the best possible solution for a given  $K$ . However, the gradual increase of deferred patients among iterations may cause a capacity cycle to increase with more than one appointment. This implies that certain capacity cycles might not be evaluated (e.g. first  $K = 16$  would have been evaluated and thereafter  $K = 18$ , so  $K = 17$  would not have been evaluated), which makes it hard to prove that the algorithm of Kortbeek et al. [37] gives an *optimal* solution (the optimal solution might for example be found when  $K = 17$  would have been evaluated). Figure 2.2 presents a visualisation of the approach as presented in Kortbeek et al. [37].

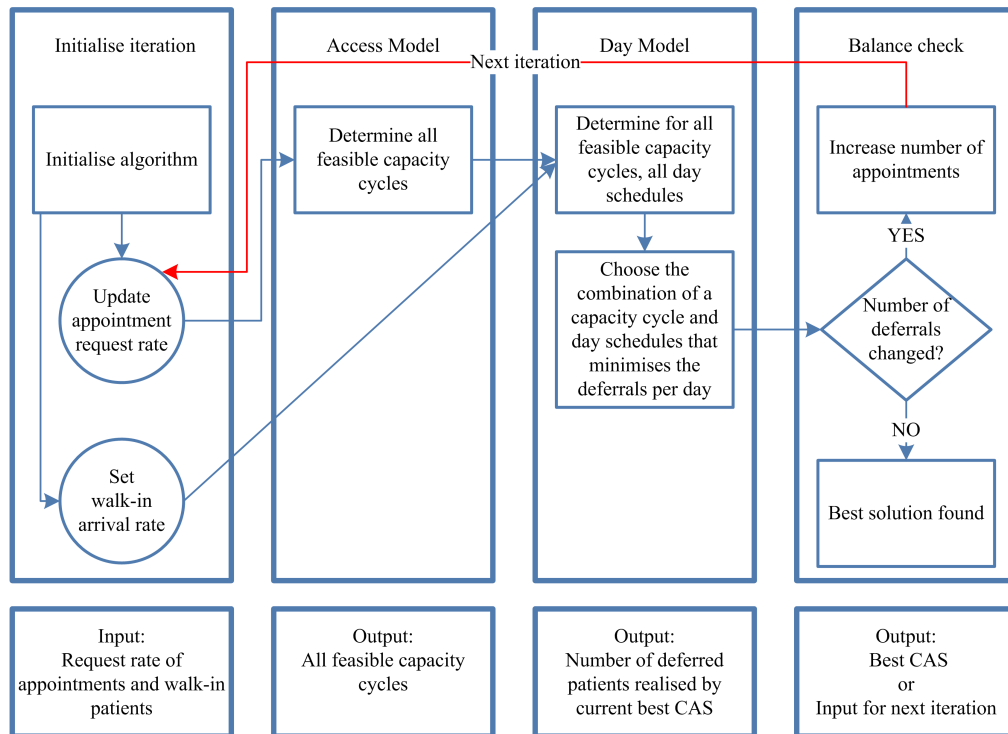


Figure 2.2: Algorithm of Kortbeek et al. [37]

## 2.4 Summary & conclusion

As discussed in Chapter 1 our aim is to develop a fast algorithm that works under the same assumptions as in [37]. Ideally, our algorithm outperforms existing scheduling techniques that can deal with appointments and walk-in patients, as we discuss in Chapter 3.

We use the algorithm of Kortbeek et al. [37] as starting point for our research to develop good appointment schedules. Their method consists of two models: one to evaluate the access process of patients with an appointment (Access model) and one to evaluate the day process of appointments and walk-in patients (Day model). Subsequently an iterative algorithm maximises the number of walk-in patients that can be served on the day of arrival while satisfying the access time service level norm for patients with an appointment request.

Finding the best CAS is done in the algorithm of Kortbeek et al. by complete enumeration. This way of working guarantees a good schedule, although this may take very long because all possible solutions have to be evaluated in each iteration.

# Chapter 3

## Literature review

In this chapter we give an overview of established theories and state of the art research in appointment scheduling. We start in Section 3.1 with a review of appointment scheduling research for outpatient facilities. We discuss methods to find good capacity cycles in Section 3.2. In Section 3.3 we give an overview of methods that can be used for making day schedules that incorporate appointments and walk-in patients. Section 3.4 ends this chapter with a summary and conclusions.

### 3.1 Appointment scheduling in outpatient clinics

Well designed appointment schedules are able to deliver timely and convenient access to healthcare for all patients. Besides that, physician idle time and patient waiting time should be minimised [24]. It is desirable to schedule patients that request for an appointment as fast as possible, since excessive access time to health services can lead to serious safety concerns [41]. Access time is measured as the time between the day of the appointment request and the day the appointment takes place. Besides timely access, long waiting times should be avoided to keep the satisfaction of patients and quality of care at a high level [46]. Waiting time is measured as the time in minutes between the time of arrival at the facility and the time of service. Earlier research at the AMC shows that a combination of appointments and walk-in results in the best performance with regard to timely access and patient preferences [38, 52]. Kopach et al. [36] discourage to allow for too many walk-in patients. They show that this can lead to a deterioration of the continuity of care, because there is a higher probability of seeing a different physician than the one the patient is used to see. This results in lower patient satisfaction and quality of care.

One of the first papers that addresses appointment scheduling in healthcare is the work of Welch and Bailey [60]. The appointment rule they describe schedules two patients in the first time slot, one patient in each time slot thereafter and no patient in the last time slot. This rule appears to perform very well in appointment scheduling [11, 28, 34, 53]. However, in the sixty years after their work, still no generally accepted way for making good appointment schedules has been found [3, 13, 20, 28]. Cayirli and Veral [10] present an extensive review of outpatient appointment scheduling research. They argue that many scholars focus on developing models for a certain hospital or case, but that the development of generally applicable appointment schedules has not been done so far. A reason why the developed models are not generally applicable in most cases, may be that problem parameters (e.g. arrival probabilities of patients) heavily affect appointment schedules and their performance [53].

The appointment system we study can be characterised as an adapted Individual Block, Fixed Interval system (IBFI). In a pure IBFI each patient has a different appointment time and these appointments are equally spaced over the day [13, 55]. Instead of only scheduling appointments, our model and the

model of Kortbeek et al. [37] also include no-shows and walk-in patients.

Hassin and Mendel [26], Kaandorp and Koole [29] and Ratcliffe et al. [48] are some examples that discuss incorporating no-shows, however these authors do not take walk-in patients into consideration. Cayirli et al. [12] present several appointment rules for making appointment schedules in outpatient clinics with no-shows and walk-in patients, which were evaluated by applying simulation and appear to perform well. However, to the best of our knowledge no paper discusses the behaviour of walk-in patients as Kortbeek et al. [37] do. In Kortbeek et al. walk-in patients get an appointment in case their service cannot start within a certain number of time slots (e.g. within half an hour). Other authors let walk-in patients wait as long as necessary for service or let them call at the start of a day for an appointment on that day, which is often referred to as same day scheduling, open access or advanced access [41]. This decreases the uncertainty in the arrival behaviour of walk-in patients as modelled by Kortbeek et al. [37].

In other businesses than healthcare, researchers study appointment scheduling techniques that deal with walk-ins as well. A revenue management approach that makes a trade-off between reservations and walk-in customers in the hotel business is discussed by Bitran and Gilbert [8] and the same is done for airlines (see for example Talluri and van Ryzin [57]) and restaurants (see for example Bertsimas and Shioda [6] or Kimes et al. [32]). However, these models cannot easily be applied to our problem. These models use prices to control access to a resource, which is generally not the leading performance measure for decision making in healthcare [24].

## 3.2 Finding good capacity cycles

Methods to determine how many appointments should be scheduled per day of the planning cycle, under the constraint that walk-in patients are served within a certain number of time slots and that patients with an appointment request are served within a certain number of days, are scarce in literature. We could not find any author, besides Kortbeek et al. [37], that determines how these appointments should be divided over the days in the planning cycle under these constraints. For determining the total number of appointments that has to be scheduled in an entire cycle, Kortbeek et al. [37] generate all capacity cycles and choose the best one. They use discrete-time queuing analysis to evaluate this access process. Kim and Giachetti [31] present a stochastic mathematical model that determines the optimal number of appointments that can be scheduled to maximise profit, while also no-shows appear and walk-in patients can enter the facility. One of the cost components they use to calculate profit is the cost for rejected patients, which is comparable to the deferred patients in the model of Kortbeek et al. [37]. However, they do not incorporate the possibility to offer a deferred patient an appointment on another day. Thereby, they assume that walk-in patients are willing to wait until the end of the day to get service. Qu et al. [47] demonstrate that the optimal percentage of time slots held open for patients that call for an appointment within 12 - 72 hours (call-in) is dependent on the ratio of the average call-in demand to the server capacity. However, in their model patients enter the facility via open-access (i.e. they get an appointment in 12 - 72 hours from the moment they request for an appointment), which reduces the scheduling problem to a pure appointment scheduling problem without the walk-in characteristic we are studying. Balasubramanian et al. [4] develop an analytic model to optimally allocate limited physician capacity in an outpatient clinic with both appointments and walk-in patients, while maximising timely access and continuity of care (i.e. building a strong or permanent relationship between a patient and a specific physician). Optimal allocation means that their model determines how many time slots should be left open for walk-in patients on a single day (so they do not make a planning for multiple days). However, this paper takes continuity of care as main performance measure, such that the probability that patients see their preferred physician is high.

No walk-in is used as in Kortbeek et al. [37], but urgent patients get an appointment on the day of the appointment request (open access). Balasubramanian et al. [4] conclude that a heuristic method that suggests the number of time slots to reserve for appointments would complement their work. However, since a schedule is made for only one day and all patients (including urgent patients) arrive by appointment, their approach reduces to an appointment scheduling problem without the walk-in property as in Kortbeek et al. [37].

### 3.3 Finding good day schedules

Appointment scheduling techniques to make day schedules can be divided in exact approaches and heuristics. Exact approaches are applied to design and optimise appointment schedules, whereas heuristics only design schedules. Heuristics can be subdivided in constructive heuristics and local search techniques. A constructive heuristic generates a solution (appointment schedule) based on some preset rules. A local search technique modifies an initial solution several times to obtain improved solutions. In the remainder of this section we discuss earlier research that uses these methods to design good day schedules.

**Exact approaches.** Several attempts are made in literature to find exact solutions for appointment scheduling problems. We take the approach as developed by Kortbeek et al. [37] as starting point of our research and our work is therefore comparable to their study. Other authors that also use exact approaches and that only differ slightly from our approach (e.g. in the behaviour of walk-in patients) are Kolisch and Sickinger [35] and Gocgun et al. [22]. The latter tries to derive properties of optimal appointment schedules, although these are refuted by Sickinger and Kolisch [53]. We describe these papers in more detail in the remainder of this section.

Kortbeek et al. [37] see appointment systems as a combination of two separate queuing systems. One system concerns the process of patients making an appointment for treatment on a certain day (Access model), whereas the other system concerns the service of patients on a specific day (Day model). They use finite time Markov chain theory to analyse the day model. Their work is an extension to the model of Creemers and Lambrecht [17], who present a similar approach as Kortbeek et al. [37] but do not consider walk-in patients. Kolisch and Sickinger [35] present a Markov Decision Process (MDP) for scheduling patients on two parallel CT-scanners. In their research scheduled outpatients arrive according to an appointment schedule, whereas inpatients and emergencies arrive at random and can thus be seen as walk-in patients. This is comparable to our study, although Kolisch and Sickinger [35] assume that emergency patients are served immediately after arrival and arriving inpatients are willing to wait until the end of the planning horizon. This is different compared to our study.

Gocgun et al. [22] conducted a study that determines per time slot which decision to take (i.e. which patient type to serve). They conclude that the optimal policy (with revenue as performance indicator) has a threshold structure. In their MDP an optimal action is determined for each time slot where a given number of inpatients and outpatients are waiting for service (i.e. determine which patient type to schedule in that time slot). They show that in the optimal policy outpatients have priority until some threshold has been reached (which represents the number of waiting inpatients). This follows from the chosen cost parameters, because the costs of an inpatient not receiving service are assumed to be much larger than the rejection costs for outpatients. Furthermore they show that a priority-based scheduling heuristic performs second-best to their optimal policy, but this heuristic approach does not give an optimal solution. However, Sickinger and Kolisch [53] question the applicability of priority-based scheduling heuristics. They mention that priority rules, where for example first all outpatients are served and thereafter all inpatients (based on Green et al. [23]), are not always optimal. This non-optimality is for example proven in one of their numerical examples with a schedule of 8 time

slots where the number of appointments to be scheduled is equal to the number of time slots available while overbooking of time slots is allowed. They show that in this setting it is optimal to book many outpatients early on a day (with some time slots overbooked), then leaving a time slot open for an inpatient and thereafter scheduling the last outpatient. This undermines the priority rule with first adjacent time slots with scheduled outpatients and next adjacent time slots reserved for inpatients.

Although models based on queuing theory are able to find optimal solutions (e.g. by complete enumeration of all possibilities), the practical application seems to decrease with the problem size [17, 37]. It is recognised that queuing models might work well for small problems, but that for larger problems the dimensions of the solution space become too large to optimally solve the problem in little time [19, 39, 40, 44]. Many authors that start with an MDP or other exact approach, identify this problem and present heuristics or approximation algorithms to overcome the curse of dimensionality (e.g. [23, 44]).

A last point of caution has to do with implementing an algorithm in practice. A study in two university hospitals argues that acceptance for computer-based decision rules in medical environments is low [35]. So a simple (or at least understandable and intuitive) algorithm to make appointment schedules would be preferable [22, 35]. Exact approaches are often highly mathematical, which can make implementation in practice hard.

**Constructive heuristics.** Because the solution space of exact approaches may grow rapidly with problem size, constructive heuristics can be used to find a good appointment schedule in little time. Authors that study appointment scheduling heuristics that incorporate appointments and walk-in patients, often use constructive heuristics and apply simulation for performance evaluation (e.g. [3, 39, 50]). Constructive heuristics are often expressed as appointment rules. Many authors conclude that the sixty years old appointment rule of Bailey and Welch [60] still performs very well. However, Cayirli et al. [13] develop an appointment rule that outperforms the rule of Bailey in the tests they performed. Klassen and Rohleder [34], Chen and Robinson [15] and Su and Shih [56] study different forms of appointment rules where patients with an appointment arrive and where a form of walk-in is allowed. Lin et al. [39] use a reduction heuristic instead of an appointment rule to make appointment schedules. We discuss these papers in more detail in the remainder of this section.

According to Gupta and Wang [24], heuristics based on appointment rules generally perform well in outpatient scheduling. As said, the appointment rule proposed by Bailey and Welch [60] performs very well in case the service time is deterministic and equal to one time slot [11, 12]. This rule prescribes that patients are scheduled at equal intervals in time slots of fixed size. In the first time slot two patients should be scheduled, one patient in each time slot thereafter and in the last time slot no appointment should be scheduled. Sickinger and Kolisch [53] show that the rule of Bailey also performs well with respect to total costs when taking into account scheduled outpatients, randomly arriving inpatients and randomly arriving emergency patients. They cannot recommend the use of this rule in case the costs of waiting or the costs of denied service are very high for randomly arriving inpatients. The difference with our study is that emergencies are served immediately and that inpatients (who are treated as walk-in patients) are willing to wait for service until the end of the day. For a setting where both patients with an appointment and walk-in patients visit the outpatient clinic and the service time is stochastic, Cayirli et al. [13] present a ‘universal appointment rule’ for the case that no-shows appear. In the tests they perform they show that their rule outperforms the rule of Bailey and that optimal day schedules (with scheduled appointments, no-shows and walk-in patients) display a dome pattern in the case that service times are stochastic [26, 29, 49]. The dome pattern is formed by appointment intervals that increase from the start of the day until a certain time slot has been reached and thereafter the appointment intervals decrease towards the end of the day. However, in their work walk-ins are assumed to arrive randomly while in our study walk-in arrivals follow a non-stationary Poisson process

with time slot dependent arrival rates. Thereby, their rule takes service of walk-ins into account by dynamically changing the length of the appointment intervals. It might for example be that the mean service time of a patient is one time slot, but that we also expect a walk-in patient to arrive in 50% of the cases in that time slot. Cayirli et al. [13] suggest then to schedule the first appointment at  $t = 1$  and the second appointment at  $t = 2.5$  to take the expected effect of an arriving walk-in patient into account. This differs also from our study, where service of a patient is assumed to take one time slot and where walk-in patients are explicitly taken into account.

Klassen and Rohleder [34] suggest that leaving time slots open for urgent patients at the beginning of a day is good for reducing waiting time, while leaving these time slots open at the end of the day results in more urgent patients being helped. In a later study these authors report that spreading urgent slots equally over the day performs best with respect to waiting time and the number of patients served [33]. Su and Shih [56] report that an alternating sequence of appointments and walk-in patients performs best, which is comparable to spreading urgent slots equally over the day. Chen and Robinson [15] study the case of patients that call at the beginning of a day for an appointment later that day. Based on their research, they argue that the first few time slots on a day should be dedicated to patients that made a call at the beginning of a day. After this period, a block should be scheduled with appointments that were made earlier and at the end of the day a block of patients that called for an appointment should be allowed again. Although patients are served on the day of their call, these patients cannot be compared to the walk-in patients we study. Because call-in patients make an appointment, they know when to come to the clinic and this avoids excessive waiting times (such that the probability of deferral is much lower than in our case).

Lin et al. [39] develop an appointment scheduling method that incorporates no-shows and patients that call for an appointment. A planner has to decide where to schedule the calling patients (i.e. sequential clinical scheduling [39]). This can be on the day of the call or on another day. They describe a heuristic approach to solve this problem, which is not based on an appointment rule. Their aim is to reduce the solution space of the MDP model they describe, by aggregating time slots and applying an approximate dynamic programming method [39] to the new instance (which has less time slots). After this step, a myopic heuristic (i.e. a heuristic that maximises immediate rewards and does not take future information into account) determines in which of the merged slots an appointment has to be scheduled. Freville and Plateau [21] confirm the positive effect of a solution space reduction algorithm on runtime performance. Although reduction might be a fruitful idea, Lin et al. [39] place an important remark on the level of aggregation: if too much aggregation is required for fast computation, far from optimal schedules can be the result.

**Local search.** Since schedules designed by constructive heuristics can be far from optimal, local search heuristics can be applied to improve these appointment schedules [9, 59]. Local search heuristics start with an initial schedule (which may be random or well chosen, dependent on the local search technique) and improve that by changing the initial schedule. This initial schedule is often the output of a constructive heuristic. Kortbeek et al. [37] mention that the model structure of the day process suggests that local search methods might be worth exploring.

A local search procedure for finding outpatient appointment schedules without walk-in patients as described by Kaandorp and Koole [29], appears to be optimal because of its multi-modularity property. We discuss their method in more detail in Section 4.4.2. Sickinger and Kolisch [53] argue that a simple neighbourhood search outperforms Tabu search for improving outpatient appointment schedules. The problem they describe is very similar to our problem of finding day schedules. Their neighbourhood search changes the initial schedule by randomly increasing or decreasing the number of appointments in a time slot. Denton et al. [18] use Simulated Annealing (SA) to improve initial schedules for outpatient surgery scheduling. They adapt their initial schedule by moving appointments forward or

backward in time. Denton et al. [18] conclude that applying SA results in substantial improvements, but that it converges slowly. They encourage future research of more advanced local search methods such as Genetic Algorithms (GA). Vanden Bosch et al. [58] report a local search procedure that swaps appointments between time slots. Their procedure was able to give an optimal solution in 85% of the studied cases for outpatient appointment scheduling. Although these last two papers do not incorporate walk-in patients, we think that the local search techniques they use are generally applicable to any (appointment) scheduling problem. All papers that test a form of local search on their constructive heuristics report that this improvement step results in better performing schedules than the schedules generated with their heuristic procedures.

### 3.4 Summary & conclusion

In the six decades after the first paper on outpatient appointment scheduling appeared, still no generally accepted way for making good appointment schedules has been found. Although many attempts are made to find good day schedules, not so many incorporate both appointments and walk-in patients. The majority of papers that incorporate walk-in patients do not take into account that walk-in patients leave the clinic if their waiting time becomes too large. Exact approaches that do so are mostly based on queuing theory and are able to find optimal solutions. However, these approaches can only find solutions for small instances because of their computational complexity. Several authors try to derive properties of optimal solutions, but a generally accepted set of properties cannot be defined. Many authors argue that queuing models might work well for small problems, but that for larger problems the dimensions of the solution space become too large to optimally solve the problem in reasonable time.

Finding good capacity cycles is less widely discussed in literature. A few authors comment on this problem, but it has to be concluded that more research is needed to find the optimal number of time slots to reserve for appointments. Models to generate appointment schedules or capacity cycles from other businesses, like the airline industry or hotel and restaurant management, cannot easily be applied to our problem since healthcare scheduling does not use prices to control access to outpatient clinics.

Constructive heuristics are proposed to find good day schedules in little time. Constructive heuristics that appear to perform well in literature might be used as a benchmark when we evaluate the algorithm we develop. No single heuristic is accepted to be the best and some papers state that the chosen scheduling approach should be adapted to the unique environment of the outpatient clinic under study. Local search heuristics seem to have a positive effect on the performance of appointment schedules generated by constructive heuristics. Several local search techniques are discussed in literature and all of them result in an increased performance compared to the initial schedules they start with. A promising way to decrease computation time is to use a heuristic or algorithm that reduces the solution space, such that less solutions have to be evaluated to get an appointment schedule.

Finally we can conclude that simple and understandable algorithms or appointment rules would be preferable, since acceptance for computer-based decision rules in medical environments is low.



## Chapter 4

# Algorithm development

In this chapter we present our appointment scheduling algorithm that combines patients with a scheduled appointment and walk-in patients. Section 4.1 presents the structure of our algorithm. In Section 4.2 we discuss heuristics to generate capacity cycles and in Section 4.3 we present heuristics to generate day schedules. In Section 4.4 we discuss local search techniques that can improve the appointment schedules generated by our heuristics. Section 4.5 ends this chapter with a summary and conclusions.

### 4.1 Algorithm structure

The structure of our algorithm is based on the model structure of Kortbeek et al. [37] that we describe in Chapter 2. This means that our algorithm consists of three parts: determining a good capacity cycle  $K$  (access process), good day schedules  $C^d$  for each day  $d$  in the planning cycle (day process), and an iterative algorithm that links the access and day process.<sup>1</sup>

The difference of our model compared to the model of Kortbeek et al. [37] is that we use heuristics to determine the capacity cycle  $K = (k^1, \dots, k^D)$  and the day schedules  $C^d$  for each day  $d$ , whereas Kortbeek et al. [37] evaluate all possible capacity cycles and day schedules. A heuristic is a method to find a solution in a structured way. Because heuristics do not evaluate all possible solutions but generally construct only a single solution, they are fast. We expect therefore that our heuristic approach leads to better performance with respect to runtime. A disadvantage of heuristics is that their solutions might deviate from optimal solutions, such that the number of deferred patients found with our algorithm is worse than found with complete enumeration as in the method of Kortbeek et al. [37].

Our algorithm starts with the initialisation of the appointment request rates and the arrival rates of walk-in patients. After this step, an iteration of our algorithm starts with a heuristic that generates a capacity cycle  $K = (k^1, \dots, k^D)$  that meets the preset access time service level norm (so we do not create *all* feasible capacity cycles as in Kortbeek et al. [37]). This access time service level norm prescribes the percentage of appointments that should be served within a preset number of days from the day of the appointment request (i.e. 95% of the appointment requests should be served within 10 days from the request). Achieving the access time service level norm is easy if we choose a high number of appointments that we schedule over the planning cycle (i.e. defer all walk-in patients such that they get an appointment). However, our aim is to minimise the number of deferred walk-in patients, which implies that we should schedule as few appointments as possible. Our algorithm takes both the request rate of appointments, and the arrival rate of walk-in patients into account for determining a capacity cycle. In Kortbeek et al. [37] only information about the appointment request rates is used for generating a capacity cycle. We present our heuristics for generating capacity cycles in Section 4.2.

---

<sup>1</sup>Note that a list of symbols can be found on page vi.

After generating a capacity cycle another heuristic generates day schedules for all days in the planning cycle. We design these day schedules in such a way that the number of deferred patients is as low as possible. We present our heuristics for generating day schedules in Section 4.3.

From our literature review it appeared that local search techniques can often improve appointment schedules that are generated by heuristics, so we also apply local search techniques to generate improved appointment schedules. We discuss local search techniques for generating capacity cycles and day schedules with improved performance in Section 4.4. Because local search may be unguided, for example in random search, we start day  $d$  in iteration  $i+1$  with the best appointment schedule found in iteration  $i$  in case the capacity on that day has not changed among subsequent iterations. By doing this, the day schedule of that day  $d$  can only be improved in upcoming iterations. In case the capacity of that day has changed, we use one of our day schedule heuristics to generate an initial schedule.

The final step in an iteration of our algorithm is to perform a balance check, which is the same as the one discussed by Kortbeek et al. [37]. We start a new iteration if no balance in the number of deferred patients has been obtained for all days in the planning cycle and the algorithm is finished if this balance is present. In Figure 4.1 we present an overview of our algorithm, which is based on the overview in [37]. The phases where we propose to use a local search technique do not necessarily have to be executed, since local search techniques only have the potential to improve the solutions found with our heuristics.

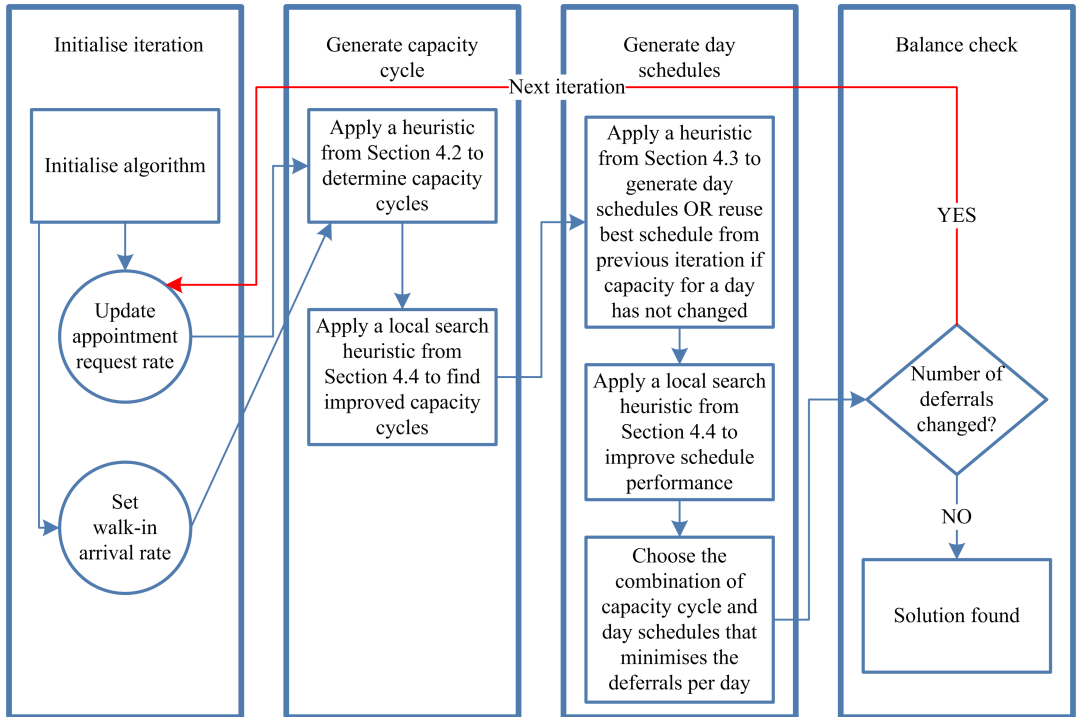


Figure 4.1: Algorithm overview

## 4.2 Heuristics for generating capacity cycles

When we want to create a good capacity cycle  $K = (k^1, \dots, k^D)$ , we first have to know how many time slots  $\Gamma$  we have to reserve for appointments over the entire length of the cycle (i.e.  $\Gamma = \sum_d k^d$ ). We argue that it is not necessary to find the best  $\Gamma$  in one iteration of our algorithm. Since in each iteration the number of deferred patients is added to the appointment request rate,  $\Gamma$  can and should be adjusted in each iteration.

For a feasible appointment schedule, the number of time slots for appointments (capacity) should be larger than the number of appointment requests (demand). We propose to use Equation 4.1 to determine an initial value for  $\Gamma$ , which serves as input for the heuristics we discuss later in this section. In Equation 4.1,  $\gamma^d$  is the appointment request rate.

$$\Gamma = \lceil \sum_{d=1}^D \gamma^d \rceil \quad (4.1)$$

The last step to generate a capacity cycle  $K = (k^1, \dots, k^D)$  is to allocate the total of  $\Gamma$  appointments over the days  $d$  in the planning cycle. It should be noted that it is not guaranteed that Equation 4.1 gives a capacity with which a capacity cycle can be generated that satisfies the access time service level norm  $S^{norm}(y)$ . The access time service level,  $S(y)$ , which is the percentage of patients with an appointment that is served within  $y$  days, should be larger than or equal to this access time service level norm  $S^{norm}(y)$ . The access time service level norm can be set by the management of the facility. Kortbeek et al. [37] present the steps to calculate the access time service level  $S(y)$ , based on the access time probability distributions. The access time service level norm can for example be  $S^{norm}(10) = 95\%$ . This means that 95% of the patients with an appointment request should be served within  $y = 10$  days after their appointment request.

If we find a capacity cycle  $K = (k^1, \dots, k^D)$  that does not meet this access time service level norm, we reserve more time slots for appointments. We proceed adding appointments until we get a feasible capacity cycle (i.e. the access time service level norm has been reached under the found capacity cycle and capacity  $\Gamma$ ). This increasing of  $\Gamma$  is done in the heuristics that we present in Sections 4.2.1 and 4.2.2. In case we use a local search technique for capacity cycles (as we discuss in Section 4.4.1), we first generate neighbouring capacity cycles before increasing the number of appointments to allocate over the cycle. This increases the probability that we find a cycle that satisfies the access time service level norm, while the cycle capacity remains low. If it appears that none of these neighbouring cycles satisfies the access time service level norm, we proceed with increasing the number of appointments that has to be allocated. Scheduling additional appointments results in less available remaining resources to serve walk-in patients. This might result in more deferred patients, which deteriorates the performance of the generated schedule. It is thus important to schedule as few appointments as possible.

In Sections 4.2.1 and 4.2.2 we propose two heuristics that can be used to find a good capacity cycle. Both heuristics are constructive and based on the idea that the workload per day, which consists of walk-in patients and patients with an appointment, should be levelled. Levelling means that all days of the planning cycle have approximately the same workload. These heuristics assume that the total number of appointments  $\Gamma$  that has to be reserved in a cycle is known and initially calculated using Equation 4.1.

### 4.2.1 Workload levelling based on mean walk-in arrival rates

Walk-in patients are preferably served on the day of their arrival, whereas patients with an appointment request can be scheduled on any day in the planning cycle as long as the access time service level norm is satisfied. We operationalise this by determining how many slots  $\psi^d$  we expect to remain free per day if we serve all walk-in patients that we expect to arrive on that day.

This heuristic iteratively allocates an appointment to the day with the highest number of expected free slots. If there is a tie, we choose for the earliest day in the week, but another allocation would also be possible. To make sure that we take the already allocated appointments into account when determining the next appointment to be scheduled, we decrease the expected number of free slots with one on the day where an appointment was added. This is done until the number of allocated

appointments  $N$  is equal to the total number of appointments  $\Gamma$  that we have to allocate in the cycle. Obviously, the number of reserved slots on a day cannot be larger than the total number of available slots  $R \cdot T$  on that day. At the end of this heuristic, we check whether the access time service level of the capacity cycle found satisfies the access time service level norm. In case this norm is violated, we increase the total number of appointments in the cycle  $\Gamma$  and execute Heuristic 1 again or we first execute the local search procedure for capacity cycles (**LS-CC**) that we describe in Section 4.4.1. We proceed until we find a feasible capacity cycle. In case we apply **LS-CC** and we find multiple cycles that satisfy the access time service level norm, we will generate day schedules for all these day schedules with one of our day schedule heuristics that we describe in Section 4.3. From these cycles with their corresponding day schedules, we select the best performing combination. Heuristic 1 gives an overview of this procedure in pseudo code.

---

**Heuristic 1** Generating a capacity cycle  $K = (k^1, \dots, k^D)$

---

```

 $\Gamma := \lceil \sum_{d=1}^D \gamma^d \rceil$ 
for  $d := 1$  to  $D$  do
     $\psi^d := R \cdot T - \sum_{t=1}^T \chi_t^d$ 
     $\psi'^d := \psi^d$ 
     $k^d := 0$ 
end for
 $N := 0$ 
while  $N < \Gamma$  do
     $\delta := \arg \max \psi'^d$ 
    if  $k^\delta < R \cdot T$  then
         $k^\delta := k^\delta + 1$ 
         $\psi'^\delta := \psi'^\delta - 1$ 
         $N := N + 1$ 
    else
         $\psi'^\delta := -BigM$ 
    end if
end while
if LS-CC is used then
    Execute LS-CC
end if
if  $S(y) < S^{norm}(y), \forall K = (k^1, \dots, k^D)$  then
     $\Gamma := \Gamma + 1$ 
    Execute Heuristic 1
end if

```

---

To clarify the working of Heuristic 1 we give an example in Table 4.1. Assume that we have to allocate  $\Gamma = 22$  appointments to a cycle with a length of  $D = 5$  days, the initial number of free time slots per day  $\psi^d$  is equal to  $(1.5, 2.0, 5.0, 7.2, 2.3)$  and there are  $T = 8$  time slots per day. We also assume that one resource  $R$  is available.<sup>2</sup>

## 4.2.2 Workload levelling based on the probability of insufficient capacity

This heuristic is comparable to Heuristic 1 in the sense that we take the arrival rates of walk-in patients into account and that we want to level the workload over all days in the planning cycle. However, in this heuristic we allocate appointments to the days that have the lowest probability  $\omega^d$  of insufficient capacity. We define the probability of insufficient capacity as the probability that the expected number

---

<sup>2</sup>These settings are the same as in the last iteration of the numerical example given in Kortbeek et al. [37].

Iteration	$\psi^d$	Capacity cycle $K = k^1, \dots, k^D$
0	(1.5, 2.0, 5.0, 7.2, 2.3)	(0, 0, 0, 0, 0)
1	(1.5, 2.0, 5.0, 6.2, 2.3)	(0, 0, 0, 1, 0)
2	(1.5, 2.0, 5.0, 5.2, 2.3)	(0, 0, 0, 2, 0)
...	...	...
21	(-0.5, -1.0, 0.0, <i>BigM</i> , -0.7)	(2, 3, 5, 8, 3)
22	(-0.5, -1.0, -1.0, <i>BigM</i> , -0.7)	(2, 3, 6, 8, 3)

Table 4.1: An example for generating a capacity cycle with Heuristic 1

of arriving walk-in patients becomes larger than the available capacity when an extra time slot would be reserved for appointments on a day  $d$ . Equation 4.2 formally represents this probability that the expected number of arriving walk-in patients  $A^d$  on day  $d$  becomes larger than the number of free time slots when an extra time slot is reserved for an appointment on that day. In each step of this heuristic we increase the number of time slots to reserve for appointments on day  $d$  by one on the day that has the lowest probability  $\omega^d$ .<sup>3</sup> Since the arrival rate of walk-in patients is Poisson distributed,  $A^d$  is Poisson distributed as well with mean  $\sum_{t=1}^T \chi_t^d$ . Since  $A^d$  follows a Poisson distribution we can compute Equation 4.2, as is shown in Equation 4.3.

$$\omega^d = \mathbb{P}(A^d \geq (R \cdot T - k^d)) \quad (4.2)$$

$$\omega^d = 1 - \sum_{i=0}^{R \cdot T - k^d - 1} \frac{\lambda^i}{i!} \cdot e^{-\lambda} \text{ where } \lambda = \sum_{t=1}^T \chi_t^d \quad (4.3)$$

Heuristic 2 continues with allocating appointments until the number of allocated appointments  $N$  is equal to  $\Gamma$ , like in Heuristic 1. The maximum number of appointments that can be allocated to a certain day is again bounded by the number of available time slots on that day. At the end of this heuristic, we check whether the access time service level of the capacity cycle found satisfies the access time service level norm. Again, we use the approach of Kortbeek et al. [37] to calculate this access time service level. In case the norm is violated, we increase the total number of appointments in the cycle  $\Gamma$  and execute Heuristic 1 again, or we first execute the local search procedure for capacity cycles (**LS-CC**) that we describe in Section 4.4.1. We proceed until we find a feasible capacity cycle. Heuristic 2 gives an overview of this procedure in pseudo code.

To clarify the working of Heuristic 2 we present a numerical example in Table 4.2. Assume again that we have to allocate  $\Gamma = 22$  appointments to a cycle with a length of  $D = 5$  days, the expected number of arriving walk-in patients per day  $A^d$  is equal to (6.5, 6.0, 3.0, 0.8, 5.7) and there are  $T = 8$  time slots per day. We also assume that one resource  $R$  is available.<sup>4</sup>

Allocated Appointments	$\omega^1, \dots, \omega^D$	$\min \omega^d$	Capacity cycle $K = k^1, \dots, k^D$
0	(0.327, 0.256, 0.012, 0.000, 0.216)	0.000	(0, 0, 0, 0, 0)
1	(0.327, 0.256, 0.012, 0.000, 0.216)	0.000	(0, 0, 0, 1, 0)
2	(0.327, 0.256, 0.012, 0.000, 0.216)	0.000	(0, 0, 0, 2, 0)
...	...	...	...
21	(0.631, 0.715, 0.577, <i>BigM</i> , 0.673)	0.577	(2, 3, 5, 8, 3)
22	(0.631, 0.715, 0.801, <i>BigM</i> , 0.673)	0.631	(2, 3, 6, 8, 3)

Table 4.2: An example for generating a capacity cycle with Heuristic 2

<sup>3</sup>This is not the exact probability that we have to defer a walk-in patient but an approximation. It is not clear in which time slot the added appointment is scheduled, which is important to evaluate the performance of a schedule.

<sup>4</sup>These settings are the same as in the last iteration of the numerical example given in Kortbeek et al. [37].

---

**Heuristic 2** Generating a capacity cycle  $K = (k^1, \dots, k^D)$ 

---

```
 $\Gamma := \lceil \sum_{d=1}^D \gamma^d \rceil$ 
for  $d := 1$  to  $D$  do
   $k^d := 0$ 
   $\omega^d := \mathbb{P}(A^d \geq (R \cdot T - k^d))$ 
end for
 $N := 0$ 
while  $N < \Gamma$  do
   $\delta := \operatorname{argmin} \omega^d$ 
  if  $k^\delta < R \cdot T$  then
     $k^\delta := k^\delta + 1$ 
     $\omega^\delta := \mathbb{P}(A^\delta \geq (R \cdot T - k^\delta))$ 
     $N := N + 1$ 
  else
     $\omega^\delta := \text{BigM}$ 
  end if
end while
if LS-CC is used then
  Execute LS-CC
end if
if  $S(y) < S^{\text{norm}}(y), \forall K = (k^1, \dots, k^D)$  then
   $\Gamma := \Gamma + 1$ 
  Execute Heuristic 2
end if
```

---

### 4.3 Heuristics for generating day schedules

In this section we discuss two heuristics to generate day schedules. A day schedule for day  $d$  is represented by  $C^d = (c_1^d, \dots, c_T^d)$ , where  $c_t^d$  represents the number of scheduled appointments at time slot  $t$ . Many authors in literature argue that using fixed appointment rules works well for the facilities or problem instances they study. Other authors in literature suggest that appointment rules cannot generally be applied, but that they only work well for the facility under study [53]. This is because in a general diagnostic facility many parameters influence the optimal location of appointments in the schedule (e.g. the arrival rate of walk-in patients or the number of time slots a walk-in patient is willing to wait for service). Different parameter configurations (i.e. another diagnostic facility) can therefore result in less performing schedules when appointment rules are applied. This is what we want to avoid, since we are looking for a generally applicable appointment scheduling algorithm.

In Section 4.3.1 we present our first heuristic to generate a day schedule. This heuristic is constructive and generates a day schedule while taking the arrival rates of walk-in patients and the available number of resources into account. We present our second day schedule heuristic in Section 4.3.2. This second heuristic does not generate an entire day schedule, but reduces the solution space by forcing some time slots to be empty. This heuristic might be of interest when applying a local search technique, such that not the entire solution space is available for generating neighbour solutions. Both heuristics assume that a feasible capacity cycle has been found already (i.e. by executing one of the heuristics that we presented in Section 4.2).

### 4.3.1 Allocate appointments to time slots with few expected walk-in arrivals

In our model walk-in patients have the characteristic that they are willing to wait a maximum of  $g$  time slots for service after their arrival. If a walk-in patient receives no service in this interval, the patient is offered an appointment on another day (the patient becomes a *deferred* patient). We exploit this characteristic for finding a day schedule. The first step of this heuristic is to determine the number of resources (where we can serve a patient) we expect to be free per time slot,  $\theta_t^d$ . This  $\theta_t^d$  depends on the time walk-in patients are willing to wait for service. It is equal to the number of available resources minus the sum of the expected walk-in arrivals in the time slots  $t - g$  until the current time slot  $t$  and the appointments that are scheduled in those time slots. We allocate an appointment to the time slot where  $\theta_t^d$  is the highest, such that expected busy time slots with respect to walk-in arrivals are left open. We do this until the number of reserved time slots for appointments is equal to  $k^d$ , and we do this for each day. Each time we allocate an appointment to a time slot, we update the value of  $\theta_t^d$ . The maximum number of appointments per time slot is bounded by the available capacity  $R$ . It should be noted that it is not our intention to present a heuristic that finds the best possible schedule. The schedule generated with this heuristic should be used as an initial schedule for the local search techniques we describe in Section 4.4. Heuristic 3 gives an overview of this procedure in pseudo code.

---

**Heuristic 3** Generating a day schedule  $C^d = (c_1^d, \dots, c_T^d)$

---

```

for  $d := 1$  to  $D$  do
  for  $t := 1$  to  $T$  do
     $\theta_t^d := \sum_{\tau=\max(t-g,0)}^t R - \chi_\tau^d$ 
     $c_t^d := 0$ 
  end for
   $N := 0$ 
  while  $N < k^d$  do
     $\delta := \arg \max_t \theta_t^d$ 
    if  $c_\delta^d < R$  then
       $c_\delta^d := c_\delta^d + 1$ 
       $N := N + 1$ 
      for  $t := \delta$  to  $\min(\delta + g, T)$  do
         $\theta_t^d := \theta_t^d - 1$ 
      end for
    else
       $\theta_\delta^d := -BigM$ 
    end if
  end while
end for

```

---

To clarify the working of Heuristic 3 we give an example in Table 4.3. Assume that we have to allocate  $k^d = 6$  appointments to a day  $d$  with 8 time slots and  $R = 1$  resource. The arrival rate of walk-in patients  $\chi_t^d$  is equal to  $(0.15, 0.30, 0.45, 0.60, 0.60, 0.45, 0.30, 0.15)$  and walk-in patients are willing to wait a maximum of  $g = 2$  time slots for service.<sup>5</sup>

### 4.3.2 Reduce the solution space by forcing time slots to be empty

In Heuristic 3 we sequentially build up complete day schedules for all days in the planning cycle. However, the heuristic that we present in this section is a reduction heuristic that does not generate a complete schedule. This heuristic determines in which time slots no appointments should be allocated.

---

<sup>5</sup>These settings are the same as in the last iteration of day 3 in the numerical example as given by Kortbeek et al. [37].

$N$	Time slot $t$ :	1	2	3	4	5	6	7	8
0	$\theta_t^d$ :	0.850	1.550	2.100	1.650	1.350	1.350	1.650	2.100
	$c_t^d$ :	0	0	0	0	0	0	0	0
1	$\theta_t^d$ :	0.850	1.550	1.100	0.650	0.350	1.350	1.650	2.100
	$c_t^d$ :	0	0	1	0	0	0	0	0
2	$\theta_t^d$ :	0.850	1.550	1.100	0.650	0.350	1.350	1.650	1.100
	$c_t^d$ :	0	0	1	0	0	0	0	1
3	$\theta_t^d$ :	0.850	1.550	1.100	0.650	0.350	1.350	0.650	0.100
	$c_t^d$ :	0	0	1	0	0	0	1	1
4	$\theta_t^d$ :	0.850	0.550	0.100	-0.350	0.350	1.350	0.650	0.100
	$c_t^d$ :	0	1	1	0	0	0	1	1
5	$\theta_t^d$ :	0.850	0.550	0.100	-0.350	0.350	0.350	0.650	0.100
	$c_t^d$ :	0	1	1	0	0	1	1	1
6	$\theta_t^d$ :	-0.150	-0.450	-0.900	-0.350	0.350	0.350	-0.350	-0.900
	$c_t^d$ :	1	1	1	0	0	1	1	1

Table 4.3: An example for generating a day schedule with Heuristic 3

We expect that this reduction heuristic is especially beneficial when we apply local search techniques. By forcing some time slots to be empty, less possible day schedules can be generated which speeds up the runtime of our algorithm. For a combination with a local search technique this means that the size of the neighbourhood decreases.

This reduction heuristic starts in the same way as Heuristic 3, where we calculate the expected number of free resources per time slot  $\theta_t^d$ . This heuristic proceeds with determining how many time slots  $z^d$  we can force to remain empty for each day. We base  $z^d$  on the number of time slots  $k^d$  that has to be reserved for appointments, multiplied with a safety factor  $\beta$ . This safety factor is needed because this heuristic might force the wrong time slots to be empty. The higher the value of the safety factor, the less time slots are forced to remain empty. We force the  $z^d$  time slots that have the lowest value of  $\theta_t^d$  to be empty on each day  $d$ . Since this heuristic does not generate a feasible day schedule, Heuristic 3 should be used afterwards to allocate appointments to the remaining free time slots. Heuristic 4 summarises this procedure in pseudo code.

---

**Heuristic 4** A reduction heuristic that forces some time slots to be empty

---

```

for  $d := 1$  to  $D$  do
   $z^d := \max(0, R \cdot T - \lceil k^d + \beta \cdot \sqrt{k^d} \rceil)$ 
  for  $t := 1$  to  $T$  do
     $\theta_t^d := \sum_{\tau=\max(t-g,0)}^t \chi_\tau^d$ 
     $c_t^d := ?$ 
  end for
   $N := 0$ 
  while  $N < z^d$  do
     $\delta := \arg \min_t \theta_t^d$ 
    if  $c_\delta^d = ?$  then
       $c_\delta^d := 0$ 
       $N := N + 1$ 
    else
       $\theta_\delta^d := \text{BigM}$ 
    end if
  end while
end for

```

---



To clarify the working of Heuristic 4 we give an example in Table 4.4. Assume that we have to reserve  $k^d = 2$  appointments on a certain day  $d$  where 8 places are available. Furthermore we assume that the arrival rate of walk-in patients  $\chi_t^d$  is equal to  $(0.30, 0.60, 1.00, 1.40, 1.40, 1.00, 0.55, 0.25)$ .<sup>6</sup> We assume that the safety factor  $\beta$  is equal to 2 in this example. The number of time slots  $z^d$  that we force to be empty would then be equal to  $T - \lceil k^d + \beta \cdot \sqrt{k^d} \rceil = 8 - \lceil 2 + 2 \cdot \sqrt{2} \rceil = 3$ .

Time slot $t$ :	1	2	3	4	5	6	7	8
$\theta_t^d$ :	0.300	0.900	1.900	3.000	3.800	3.800	2.950	1.800
$c_t^d$ :	?	?	?	0	0	0	?	?

Table 4.4: An example for generating a reduced day schedule solution space with Heuristic 4

## 4.4 Local search for appointment schedule improvement

Heuristics generally give non-optimal solutions. Therefore we use local search techniques to find appointment schedules with an improved performance, compared to the appointment schedules found with our capacity cycle and day schedule heuristics. A local search technique starts with an initial solution and makes small changes to this initial solution, such that a *neighbour* solution is created. By doing this many times, many neighbour solutions are created. This increases the probability of finding a schedule with increased performance. The local search techniques we describe in Sections 4.4.1 and 4.4.2 take the solutions of our heuristics from Section 4.2 and 4.3 as initial solution.

### 4.4.1 Generating neighbouring capacity cycles

We propose to use a simple swapping procedure to find neighbour solutions of the capacity cycle found with one of the heuristics that we presented in Section 4.2. We denote this swapping procedure by **LS-CC**. Instead of only working with the capacity cycle as generated by one of those heuristics, we increase the solution space by adding some neighbouring capacity cycles. In our swapping procedure the total number of appointments  $\Gamma$  to allocate in the planning cycle remains constant. We swap  $n$  appointments between two days (i.e. we add  $n$  appointments to a certain day and subtract  $n$  appointments on another day). In the worst case this means that we have to evaluate  $n \cdot (D \cdot (D - 1))$  additional capacity cycles when the cycle length equals  $D$ . We expect that the capacity cycle heuristics from Section 4.2 give good initial capacity cycles, such that the value of  $n$  can remain small (e.g. at most 2) to find good neighbouring capacity cycles. Generating and evaluating many neighbour solutions slows down our algorithm, but we expect that the probability to find a good final appointment schedule increases since more capacity cycles, and thus appointment schedules, are evaluated.

### 4.4.2 Local search for day schedules

For the day schedules generated with the heuristics that we presented in Section 4.3 we expect that their performance with respect to the number of deferred walk-in patients can also be improved. Therefore we propose to use local search techniques to find day schedules with a better performance. The total number of allocated appointments per day  $k^d$  is constant during our local search procedure. This ensures that neighbouring day schedules found with one of the local search techniques we propose, are still feasible with respect to the access time service level norm.

<sup>6</sup>These settings are the same as in the last iteration of day 1 in the numerical example as given by Kortbeek et al. [37].

We would like to use well known local search techniques with a good performance. However, from literature it appears that no single local search technique is dominant nowadays [27]. Bianchi et al. [7] argue that not much research has been done to compare different local search heuristics for the same problem. They give a short overview of papers that do compare local search heuristics and among the winners are tabu search and Genetic Algorithms (GA), but never Simulated Annealing (SA). Sickinger and Kolisch show that a simple random search outperforms tabu search for making appointment schedules. Kaandorp and Koole [29] present a local search technique for an appointment scheduling problem that always gives an optimal solution. Their model assumptions are comparable to our assumptions, although there are some differences. In their model no unscheduled patients can be served and they use exponentially distributed service times. Now we give a short description of these five local search techniques.

**Simulated Annealing** is based on the controlled cooling process of metals to decrease the number of defects in the metal. To find good day schedules, SA starts with a random initial schedule. In every step of the annealing procedure a neighbour schedule is evaluated. In a neighbour schedule a free time slot and a time slot reserved for an appointment are exchanged. If this exchange results in an improved schedule, this schedule is accepted. If this exchange does not result in an improved schedule, the schedule is accepted with a certain probability. This makes sure that not only schedules with improved solutions are accepted, such that escapes from local optima are possible. The acceptance probability is initially high, but decreases as the algorithm proceeds. This makes SA initially a random search method where good and bad solutions are accepted, but at the end of the process SA acts as a local search method that only accepts improved solutions.

**Tabu search** starts with a well chosen initial day schedule. From this initial schedule, all neighbour schedules are evaluated. The neighbour schedule with the best objective value (i.e. the lowest number of deferred walk-in patients) is chosen. In a neighbour schedule a free time slot and a time slot reserved for an appointment are exchanged. The last  $n$  exchanges are stored in a *tabu* list. Exchanges in this list are not allowed to be executed again to generate new neighbour schedules. Neighbour schedule generation terminates when no improvements are being found, after a fixed number of evaluations or when all neighbours are in the tabu list.

**Random search** can start with a well chosen initial schedule, but this is not necessary. From the initial schedule, a neighbour schedule is evaluated. In this neighbour schedule a randomly chosen free time slot and a randomly chosen time slot reserved for an appointment are exchanged. If the performance of this neighbour schedule is better than the initial schedule, we proceed generating new neighbour solutions based on this improved schedule. Neighbour schedule generation proceeds until a schedule is evaluated that does not result in an increased performance or after a certain number of neighbour schedules have been generated.

**Genetic Algorithms** are based on the evolution process that can be observed in nature, like survival of the fittest. As a local search heuristic, first an initial population of day schedules is generated. From this population the  $\zeta$  best performing schedules are selected. Based on these schedules new schedules are generated by mutation and recombination, which form a new population. Mutation is done by randomly exchanging a free time slot with a time slot reserved for an appointment. Recombination is done by exchanging two parts (containing multiple time slots) of a day schedule, such that a new day schedule is generated. From this new population again the  $\zeta$  best performing schedules are selected. This process proceeds until a certain number of generations have been evaluated.

**Kaandorp and Koole** present a swapping algorithm that always gives the optimal appointment schedule for the problem they study. They start with some initial schedule (might be random) and swap appointments in this schedule in a structured way. For their swapping procedure, Kaandorp and Koole [29] define the following vectors:

$$\mathcal{V}^* = \begin{pmatrix} u_1, \\ u_2, \\ \vdots \\ u_{T-1}, \\ u_T \end{pmatrix} = \begin{pmatrix} (-1, 0, \dots, 0, 1), \\ (1, -1, 0, \dots, 0), \\ \vdots \\ (0, \dots, 1, -1, 0), \\ (0, \dots, 0, 1, -1) \end{pmatrix}$$

Furthermore they define the set  $\mathcal{U}$ , which is a strict subset of  $\mathcal{V}^*$  ( $\mathcal{U} \subsetneq \mathcal{V}^*$ ). As neighbourhood of schedule  $x$  they take all vectors of the form  $x + \nu_1 + \dots + \nu_k$  with  $\nu_1, \dots, \nu_k \in \mathcal{V}^*$  such that  $x + \nu_1 + \dots + \nu_k \geq 0$ . A vector  $u_t$  can be interpreted as moving an appointment from time slot  $t$  to time slot  $t - 1$ . If a schedule is found with an improved performance, this schedule is set as the initial schedule and the local search method starts again. This procedure is repeated until a schedule is found that cannot be improved by adding one of the vectors  $\sum_{\nu \in \mathcal{U}} \nu$ . Kaandorp and Koole [29] prove that this approach results in optimal schedules. They use other performance measures than we do for day schedules, but since performance measures are only used to evaluate day schedules and not to generate them, we expect that this has no influence on the applicability of their local search technique in our algorithm. They also state that this approach takes a lot of time, but that a reduced form of their algorithm (only use solutions of the form  $x + u_t$ ) is fast and results in good schedules.

The most important characteristic of the local search technique that we want to apply is that the technique is able to generate improved appointment schedules in little time. The iterative character of our algorithm, as presented in Figure 4.1 in Section 4.1, ensures that our algorithm generates many day schedules before it finds a final appointment schedule. This holds also for the last iteration of our algorithm: the capacity cycle found in the last iteration cannot be different from the capacity cycle in (at least) the second to last iteration, since otherwise there would be no balance in deferred patients per day. Therefore the time to find better neighbour solutions with a local search technique can be small. In Table 4.5 we give a qualitative comparison of the local search heuristics that we evaluated for our study.

Local search technique	Characteristics
Simulated Annealing	+ Can converge to a global optimum
	+ Easy to implement
	- Performance dependent on parameter choices
	- Converges slowly
Tabu search	+ Easy to implement
	+ Can escape from local optima
	- Performance dependent on parameter choices
	- Outperformed by simple random search [53]
Random search	+ Easy to implement
	+ Good performance for a similar problem [53]
	- Search is unguided, outcomes are unpredictable
Genetic Algorithms	+ Relatively easy to implement
	- Search space may grow rapidly
Kaandorp and Koole	+ Can converge to a global optimum [29]
	+ Relatively easy to implement
	+ Smaller (non-optimal) neighbourhood gives good results [29]
	- Search space may grow rapidly [29]

Table 4.5: Comparison of local search heuristics

SA can find a global optimum, but its performance strongly depends on the chosen parameter values and it can take long to find better solutions than the initial solution. Thereby, SA was never among the

best performing local search techniques in the review of Bianchi et al. [7]. In their review they assessed the performance of several local search techniques for several scheduling problems. Our comparison in Table 4.5 shows that tabu search, random search, GA's and the approach of Kaandorp and Koole [29] seem the most promising with respect to implementability and the time to find good neighbour solutions. However, tabu search is outperformed by a simple random search in an earlier study on appointment scheduling [53]. Therefore we choose to work with random search (denoted by **LS-RS**), a genetic algorithm (denoted by **LS-GA**) and the approach of Kaandorp and Koole [29] (denoted by **LS-KK**). These three local search techniques are relatively easy to implement and are able to generate good solutions.

## 4.5 Summary & conclusion

In this chapter we developed the structure of our algorithm. Our algorithm is based on the model structure as presented in [37]. Instead of using complete enumeration to generate capacity cycles and day schedules, we propose heuristics and local search techniques to find a good appointment schedule fast. In a good appointment schedule few walk-in patients are deferred and the access time service level norm is satisfied for patients with an appointment request.

Our capacity cycle generating heuristics aim to level workload (consisting of scheduled appointments and walk-in patients) among days in the planning cycle. Our first day schedule heuristic allocates appointments to time slots with few expected walk-in patients. The second day schedule heuristic is a reduction heuristic, that forces some time slots to be empty. This might be beneficial when using local search techniques, since the size of the solution space decreases and that may decrease runtime.

The performance of heuristics can often be improved by using local search techniques. From literature it appears that no single local search technique is dominant in appointment scheduling. Based on reviews in literature we select Random Search, Genetic Algorithms and the local search technique of Kaandorp and Koole [29] as promising techniques to use in improving day schedules. For capacity cycle generation we use a swapping procedure that generates neighbouring capacity cycles based on an initial cycle. In Chapter 5 we will assess the performance of our heuristics and the selected local search techniques.

# Chapter 5

## Algorithm testing

In this chapter we compare the performance of our heuristics and local search techniques as presented in Sections 4.2, 4.3 and 4.4 with each other and with the performance of the algorithm of Kortbeek et al. [37]. The purpose of these tests is to identify the best combination of heuristics and local search techniques that we can use for our case study in Chapter 6 and that ultimately can be used in practice. We start with outlining our test approach in Section 5.1. In Section 5.2 we give an overview of our test settings and in Section 5.3 we present the test results. We end this chapter with a summary and conclusions in Section 5.4.

### 5.1 Approach

We propose to start testing with our capacity cycle generating heuristics, while using complete enumeration for making day schedules. In this way, we can assess the performance of our capacity cycle heuristics, without the disturbances in performance that might be caused by day schedule generating heuristics (we evaluate all possible day schedules for our heuristic capacity cycles, so the best possible day schedules are always chosen). We compare the outcomes of these tests with the outcomes as generated by the algorithm of Kortbeek et al. [37], so with complete enumeration for both capacity cycle and day schedule generation. After these tests, we can assess which capacity cycle generating heuristics perform well. The main criterion here is the number of deferred patients. A criterion that is of less interest for small instances, but that might become important for practical problems, is runtime. Therefore we also measure the runtime of all combinations.

Secondly we test all combinations of capacity cycle heuristics, day schedule heuristics and local search techniques. From these tests, we select the best performing combinations of heuristics and local search techniques. We use these combinations in our case study. We select the combinations that perform well with respect to the number of deferred patients. In case many combinations appear to perform well, we also take runtime into consideration as selection criterion. Table 5.1 gives an overview of all possible heuristics, local search techniques and exact techniques that we can use to generate appointment schedules.

Capacity Cycle Generation		Day Schedule Generation	
Method	Local Search Technique	Method	Local Search Technique
Complete enumeration ( <b>CE</b> )	Swapping for capacity cycles ( <b>LS-CC</b> )	Complete enumeration ( <b>CE</b> )	Random search ( <b>LS-RS</b> )
Heuristic 1 ( <b>H1</b> )		Heuristic 3 ( <b>H3</b> )	Genetic algorithm ( <b>LS-GA</b> )
Heuristic 2 ( <b>H2</b> )		Heuristic 4 ( <b>H4</b> )	Kaandorp and Koole [29] ( <b>LS-KK</b> )

Table 5.1: Overview of test combinations

## 5.2 Test settings

In this section we present the settings that we used in our tests. In Section 5.2.1 we present the different parameters and parameter values of the problem instances we test for. Heuristic 4 and the local search techniques we use in our tests are parameter dependent. This means that the chosen values for the parameters influence the performance of Heuristic 4 and the local search techniques. In Section 5.2.2 we give an overview of the parameters and their values in our tests.

### 5.2.1 Parameters test instances

For the combinations of capacity cycle heuristic, day schedule heuristic and local search techniques we perform tests for several problem instances. In Table 5.2 we give an overview of the test parameters and the values of these parameters we test for.

Parameter	Description	Base case	Low value	High value
$R$	Number of available resources	1	–	–
$D$	Length of the planning cycle	5	–	–
$T$	Total number of available time slots per day	8	–	–
$g$	Maximum waiting time in time slots that a walk-in patient accepts for service	2	–	4
$q$	No-show probability	0.15	0	–
$(y, S^{norm}(y))$	Access time service level norm: fraction of jobs with access time not greater than $y$ is at least $S(y)$	(10, 95%)	(5, 95%)	(15, 95%)

Table 5.2: Overview of test parameters

From Table 5.2 it is clear that we only test for small instances with 8 time slots per day and a planning cycle of 5 days. We have explicitly chosen to do this because we are now able to compare the performance of our heuristics to the performance of the algorithm of Kortbeek et al. [37]. The algorithm of Kortbeek et al. [37] is based on complete enumeration, so the schedules found with their approach are the best we can compare with. When we would increase the number of time slots, the length of the planning cycle or the number of resources, the algorithm of Kortbeek et al. [37] is not able to generate a solution anymore. This is because of limitations on runtime and computer memory. Since we want to compare the solution found with our algorithm to the solution found by complete enumeration, we perform the tests in this chapter only on small instances. We argue that testing on a small instance is sufficient to reject a number of possible heuristics. If the performance of a combination of heuristics deviates a lot from complete enumeration on small instances, there is a high probability that this combination would give a bad performance for a large instance also.

We adjust the maximum waiting time a walk-in patient is willing to wait,  $g$ , since this parameter is expected to be dependent on the kind of facility. Earlier research has shown that walk-in patients are willing to wait before they receive service [52]. We test with two values, such that we can assess what the influence is of a larger or smaller maximum waiting time for walk-in patients. Another parameter we adjust is the no-show probability of patients with an appointment. Literature reports varying values of this no-show rate, but we found that most values are between 10% and 20% [11, 28, 58]. We test therefore for a maximum no-show rate of 15%. We think that it is also interesting to see how our heuristics perform when every patient shows up, because then the workload of the facility might become very high. We also adjust the access time service level norm, since this norm can be set by

the management of the facility and thus can be different among clinics. The base case access time service level norm is set on serving 95% of the appointment requests within 10 days from their request, which corresponds with patient preferences in earlier research [52]. However, we think that it is also interesting to see how our heuristics perform when this norm is tighter or more loose. Therefore we also test for an access time service level norm where 95% of the appointments should be served within 5 and 15 days respectively.

Besides the parameters as presented in Table 5.2, we also test our heuristics for different appointment request rates and walk-in arrival rates. We test for three theoretical patterns of the walk-in arrival rates as displayed in Figures 5.1, 5.2 and 5.3. The actual appointment request rates and walk-in arrival rates for pattern 1, 2 and 3 can be found in Appendix B. Pattern 1 is equal to the walk-in arrival rates as used in the numerical example of Kortbeek et al. [37] and pattern 2 and 3 are new patterns. These new patterns are theoretical and might not reflect practice. However, we think that testing for many different patterns is the best way to assess the performance of our heuristics. In Figure 5.4 we display the three appointment request rate patterns we test for, where pattern 1 is equal to the appointment request rate pattern as used in the numerical example of Kortbeek et al. [37] and pattern 2 and 3 are new patterns.

From the test parameters in Table 5.2, we test for all combinations of base case, low value and high value. This gives 12 parameter configurations in total. On each of these configurations we perform tests for patterns 1 to 3. In total we thus test for 36 instances per combination of heuristics. In our opinion testing for many problem instances is necessary to show that our heuristics are able to generate good appointment schedules under different parameter settings. In case our heuristics generate appointment schedules with good performance for all different tests, the general applicability of our heuristics to any facility is expected to be larger. In Table 5.3 we give an overview of the combinations of appointment request rate patterns and walk-in arrival rate patterns we test for.

Combination	Appointment request rate pattern	Walk-in arrival rate pattern
P1	Pattern 1	Pattern 1
P2	Pattern 2	Pattern 2
P3	Pattern 3	Pattern 3

Table 5.3: Combinations of appointment request rate patterns and walk-in arrival rate patterns

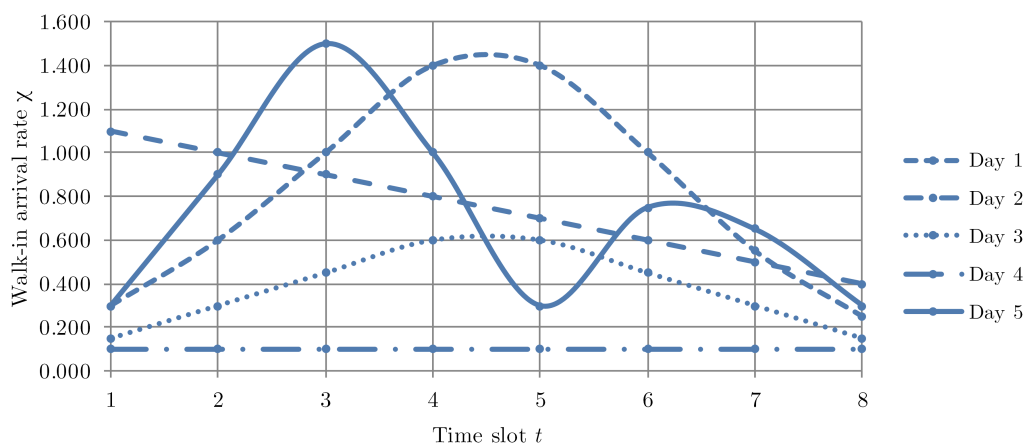


Figure 5.1: Walk-in arrival rates pattern 1 (Kortbeek et al. [37])

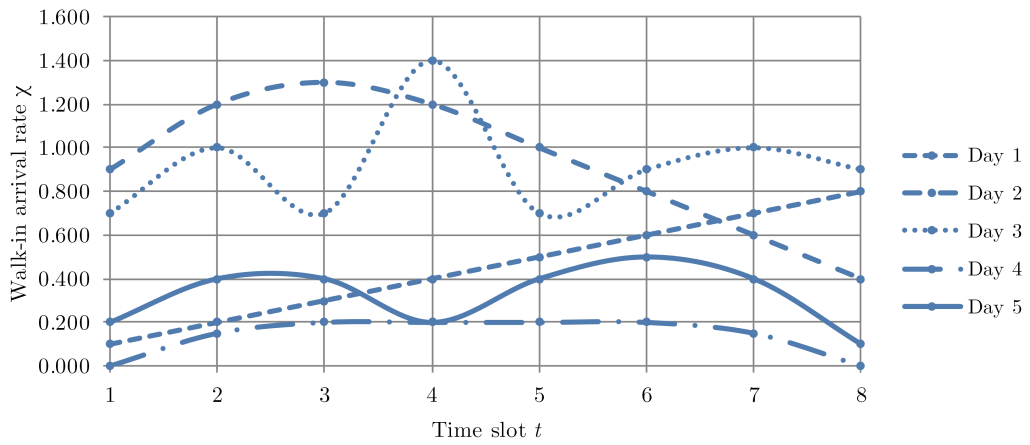


Figure 5.2: Walk-in arrival rates pattern 2

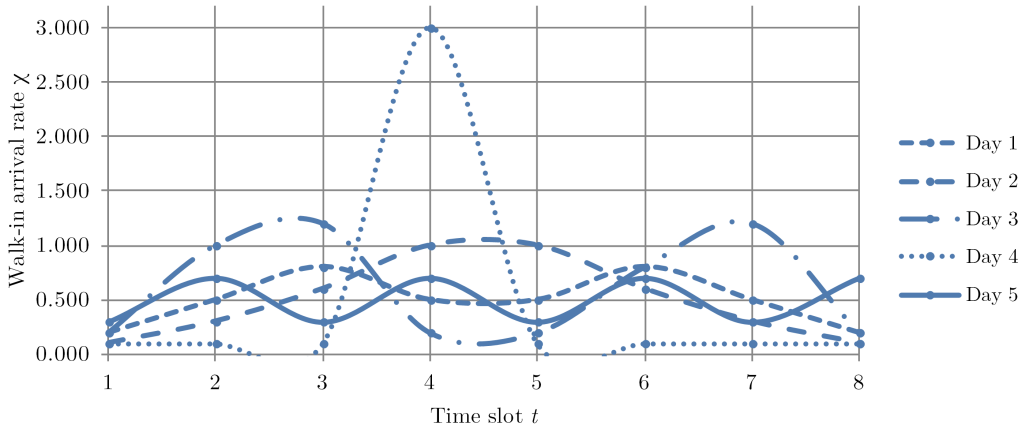


Figure 5.3: Walk-in arrival rates pattern 3

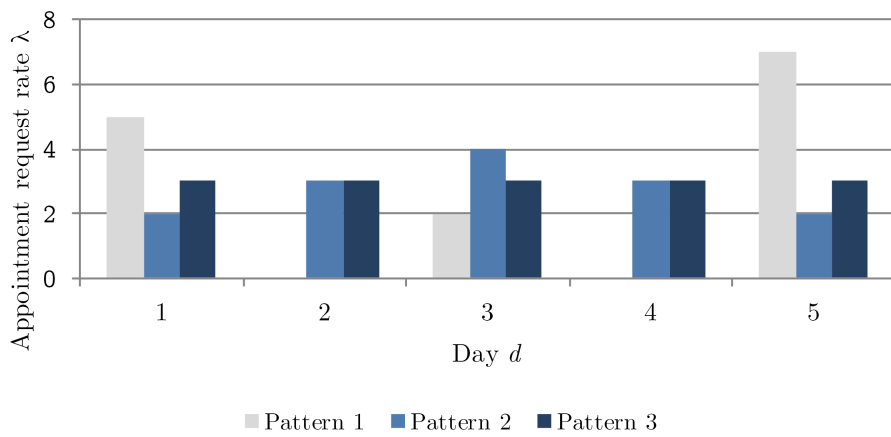


Figure 5.4: Appointment request rates



## 5.2.2 Parameters Heuristic 4 and local search techniques

Heuristic 4 and the local search techniques we use in our tests are parameter dependent. This means that the chosen values for these parameters influence the performance of Heuristic 4 and the local search techniques. Therefore we shortly discuss the chosen parameter values in this section.

**Heuristic 4.** The parameter we have to choose in our reduction heuristic for day schedules is the safety factor  $\beta$ . The value of  $\beta$  influences the number of time slots that this heuristic forces to be empty. The higher  $\beta$ , the lower the reduction in solution space. We choose to use a value for  $\beta$  of 2. This means that we increase the arrival rate of appointment requests with 2 times its standard deviation (in case the appointment request rate is Poisson distributed). More details about Heuristic 4 can be found in Section 4.3.2. The value of  $\beta$  we choose is often used in inventory management literature when for example calculating safety stocks (e.g. [54]).

**Swapping for capacity cycles.** The parameter to choose here is the maximum deviation  $n$  from the capacity cycle as generated by one of our capacity cycle generation heuristics. We choose this deviation  $n$  to be 2. This means that from the initial capacity cycle, all neighbouring capacity cycles are generated (based on the initial capacity cycle) where we subtract 1 or 2 appointments on one day and add these appointments on another day.

**Random search.** The parameter to choose here is the number of neighbouring day schedules to generate. We set this parameter to 10, such that we generate 10 randomly modified day schedules. As initial schedule we take the day schedule as generated by one of our day schedule heuristics. If a modification results in an improved schedule, we accept this schedule and proceed with modifying this schedule, until 10 schedules are generated.

**Genetic Algorithm.** The parameters to choose here are the population size, the number of mutations and recombinations and the number of generations. We start with an initial schedule generated by one of our day schedule generating heuristics. From this schedule we make a population of 10 day schedules. In this population, 5 schedules are based on the initial schedule on which we apply a random swap (mutation) and the other 5 schedules are based on a random swap in combination with exchanging the first half of time slots with the second half of time slots (recombination). We evaluate these 10 schedules and select the 4 best performing day schedules. These 4 schedules are the initial schedules of a new generation. For each of these initial schedules we apply the same procedure as just described. We proceed until we have evaluated the schedules of 2 generations (excluding the initial generation).

**Kaandorp and Koole.** In this method no parameter has to be chosen, but the choice between complete execution of the method of Kaandorp and Koole [29] or an implementation with a smaller neighbourhood has to be made. As Kaandorp and Koole [29] indicate, using their full method with a very large neighbourhood results in an optimal schedule but takes hours to calculate. They also indicate that using a small neighbourhood is fast and able to give very good results. Therefore we use a small neighbourhood, where new schedules are created (based on the initial schedule) by moving at most one appointment to an earlier or later time slot.<sup>1</sup>

In general we can conclude that we use relatively small neighbourhoods for our local search heuristics. One could argue that small neighbourhoods might deteriorate the performance of a solution, but Kaandorp and Koole [29] show that a small neighbourhood can also result in good performance. Thereby, our algorithm is iterative, which means that our local search techniques are executed again in each iteration. This iterative character would lead to an explosion of runtime in case we would use large neighbourhoods. However, we expect that these iterations have a positive effect on performance, because many new solutions are evaluated in each iteration of our algorithm.

---

<sup>1</sup>Notice that this is not the same as described by [29], where appointments could only be moved to an earlier time slot. We include the possibility to move an appointment to a later time slot, because we expect that this results in an increased performance while it is still relatively fast.

## 5.3 Results

In this section we discuss the performance of our heuristics. The performance measures we use are the number of deferred patients and algorithm runtime. In Section 5.3.1 we present the performance of our capacity cycle generating heuristics, with complete enumeration for the day schedules. In Section 5.3.2 we present the performance of our capacity cycle heuristics in combination with a day schedule generating heuristic and a local search technique.

### 5.3.1 Performance capacity cycle heuristics

In this section we present the tests we performed on our capacity cycle generating heuristics and the swapping procedure for capacity cycles (**LS-CC**). In these tests we use complete enumeration to generate day schedules. We use the walk-in arrival rate patterns and appointment request rates as displayed in Figures 5.1, 5.2, 5.3 and 5.4. Per capacity cycle method, we test for 36 instances (3 patterns with 12 instances per pattern). In Appendix C we present a complete overview of the tests we performed on our capacity cycle generating methods. All tests we refer to in this section, can be found in that appendix. We give a summary of these results in Table 5.4.

Capacity cycle method	Day schedule method	Fraction of walk-in patients served on day of arrival	Average number of deferred patients	Average deviation from CE+CE	Maximum deviation from CE+CE	Same solution as CE+CE	Average runtime (seconds)	Average runtime (minutes)
<b>CE</b>	<b>CE</b>	74.04%	5.598	-	-	-	28892.7	481.55
<b>H1</b>	<b>CE</b>	73.67%	5.678	1.43%	5.52%	39%	3.0	0.05
<b>H2</b>	<b>CE</b>	73.73%	5.666	1.22%	4.99%	31%	3.1	0.05
<b>H1+LS-CC</b>	<b>CE</b>	73.95%	5.618	0.36%	4.99%	72%	33.2	0.55
<b>H2+LS-CC</b>	<b>CE</b>	73.94%	5.620	0.40%	4.99%	69%	33.0	0.55

Table 5.4: Summary of performance capacity cycle generating heuristics

From Table 5.4 we see that all combinations of capacity cycle generating heuristics and complete enumeration to generate day schedules are able to come up with a feasible appointment schedule within one minute runtime on average. However, not all combinations achieve the same performance with respect to the number of deferred patients. As expected, when applying the model of Kortbeek et al. [37] (**CE+CE**) we find the least number of deferred patients. This corresponds with the highest fraction of walk-in patients served on the day of their arrival. Under complete enumeration we serve on average 74.04% of the walk-in patients on the day of their arrival, while our best performing combination of heuristics (i.e. **H1+LS-CC+CE**) serves on average 73.95% of the walk-in patients on the day of their arrival. This difference is very small, which means that Heuristic 1 in combination with the swapping procedure (**LS-CC**) can find cycles that are close to the cycles found under complete enumeration.

Heuristic 1 and Heuristic 2 are faster when the swapping procedure for capacity cycles is not used. This is according to our expectations, since the swapping procedure generates additional capacity cycles. For all these cycles day schedules have to be generated and those combinations have to be evaluated, which takes time. However, more capacity cycles are evaluated so the probability to find a better solution than that would be found without the swapping procedure increases. From Table 5.4 we observe indeed that using the swapping procedure for capacity cycles results in a higher fraction of the walk-in patients served on the day of their arrival. Similarly, using the swapping procedure results in increased performance with respect to the average number of deferred patients and the number of times that our heuristics obtain the same performance as under complete enumeration.

Although our capacity cycle generating heuristics in combination with the swapping procedure deviate less than 0.5% from complete enumeration on average, these combinations defer in the worst case

4.99% more walk-in patients than with complete enumeration. From our detailed results in Appendix C (tests 125 and 161) it appears that this worst case performance is obtained for the same test setting. In that setting we use arrival rate pattern 2 (Figure 5.2), walk-in patients are willing to wait a maximum of 2 time slots, the no-show percentage is 15% and 95% of the patients with an appointment request need to be served within 5 days from their request. Although this worst case appears with pattern 2, the same setting with pattern 1 and 3 also gives a relatively large deviation from complete enumeration. It appears that decreasing the no-show percentage to 0% while keeping all other settings the same (test 127) results in a deviation of 1.25%. Days in a planning cycle may have different appointment request rates, so the number of no-shows might also be different among days. However, our capacity cycle generating heuristics do not take no-shows into account. This may lead to capacity cycles that deviate from the cycles generated with complete enumeration. This is what we observe in the capacity cycle found by complete enumeration, which has a capacity cycle with one scheduled appointment less than in the Cyclic Appointment Schedule (CAS) found with **H1+LS-CC+CE** and **H2+LS-CC+CE**. However, the combination of **H1+LS-CC+CE** finds the same CAS as complete enumeration when the maximum number of time slots that walk-in patients are willing to wait is increased to 4 (test 130) and when the access time service level norm is increased to serving 95% of the patients with an appointment request within 10 days from their request (test 121). We expect that this improved performance is caused by using more relaxed settings: when walk-in patients and patients with an appointment request are willing to wait longer, we can exploit this relaxed behaviour by letting them wait during busy periods of the day (walk-in patients) or week (patients with an appointment request). For combination **H2+LS-CC+CE** the worst case behaviour also improves under these relaxed settings, however still deviations from complete enumeration of 1.39% and 0.16% are found when we relax the waiting behaviour of walk-in patients and patients with an appointment request (test 157 and 166 respectively). This suggests that having no-shows or a low allowed waiting time for walk-in patients makes it more difficult to generate an appointment schedule, but that the main deterioration is caused by a tight access time service level norm. Besides the average deviation from complete enumeration, we see that **H1+LS-CC+CE** outperforms **H2+LS-CC+CE** with the swapping procedure on the number of times that the same solution is found as under complete enumeration. This together with the worse case behaviour of both combinations implies that Heuristic 1 with the swapping procedure is better than Heuristic 2 with the swapping procedure.

One might expect that Heuristic 2 should outperform Heuristic 1, since Heuristic 1 is only based on *mean* walk-in arrival rates whereas Heuristic 2 is based on the *probability* of insufficient capacity on a day. This is what we observe (on average) when we do not use the swapping procedure (**LS-CC**). However, for some situations Heuristic 1 outperforms Heuristic 2. We analysed such a situation (tests 49 and 85, base case test instance). Recall that Heuristic 1 adds appointments to the day with the highest number of free slots, while Heuristic 2 adds appointments to the day with the lowest probability of insufficient capacity. In case that for example two days have the same number of free time slots, Heuristic 1 allocates the appointment to the earliest day of these two. At the end of the second iteration of our algorithm for these instances, Heuristic 1 allocates 5 appointments to day 1 and 5 to day 5, whereas Heuristic 2 allocates 4 appointments to day 1 and 6 to day 5 (day 2, 3 and 4 are the same for both heuristics). In this iteration a difference in cycles arises, which also results in a difference in the number of deferred patients. This suggests that allocating relatively many appointments to day 1 seems profitable, although the total number of expected walk-in patients on day 1 is higher than on day 5 (3.6 versus 2.6 respectively). Recall from Figure 5.2 that day 1 has a monotonically increasing walk-in arrival rate, whereas the walk-in arrival rate pattern on day 5 consists of two peaks that have their maxima around time slots 2 and 6. Although day 5 has a lower number of expected walk-in arrivals than day 1, it is apparently harder to schedule appointments to day 5 because of the walk-in arrival rate

pattern. Because Heuristic 1 allocates to the earliest of those days in case of a tie, it may benefit from a walk-in arrival rate pattern on day 1 in which appointments can easily be scheduled. Because the total expected walk-in arrival rate on that day might be higher, Heuristic 2 always allocates to the day with the lowest probability of insufficient capacity. In our example it appears that it is difficult to allocate appointments on day 5 (because of the walk-in arrival pattern), which results in a bad performance of Heuristic 2.

### 5.3.2 Performance day schedule heuristics

The next step of our approach is to test Heuristics 1 and 2 in combination with the swapping procedure for capacity cycles, our day schedule generating heuristics and the local search techniques for day schedules. Again we use the walk-in arrival rate patterns and appointment request rates as displayed in Figures 5.1, 5.2, 5.3 and 5.4. Per combination of heuristics, we test again for 36 instances. In Appendix D we present a complete overview of the tests we performed to select a good combination of capacity cycle generating heuristic, day schedule heuristic and local search technique. All tests we refer to in this section, can be found in that appendix. We give a summary of these results in Table 5.5.

Capacity cycle method	Day schedule method	Fraction of walk-in patients served on day of arrival	Average number of deferred patients	Average deviation from CE+CE	Maximum deviation from CE+CE	Same solution as CE+CE	Average runtime (seconds)	Average runtime (minutes)
<b>CE</b>	<b>CE</b>	74.04%	5.598	-	-	-	28892.7	481.55
<b>H1</b>	<b>H3+LS-RS</b>	73.63%	5.687	1.58%	5.70%	31%	2.1	0.03
<b>H1</b>	<b>H4+LS-RS</b>	72.93%	5.837	4.27%	11.99%	6%	2.3	0.04
<b>H1</b>	<b>H3+LS-GA</b>	73.62%	5.690	1.65%	7.18%	36%	10.1	0.17
<b>H1</b>	<b>H4+LS-GA</b>	73.31%	5.755	2.81%	10.94%	28%	31.7	0.53
<b>H1</b>	<b>H3+LS-KK</b>	72.85%	5.854	4.58%	15.13%	25%	1.8	0.03
<b>H2</b>	<b>H3+LS-RS</b>	73.72%	5.667	1.23%	4.99%	28%	2.2	0.04
<b>H2</b>	<b>H4+LS-RS</b>	73.24%	5.772	3.10%	11.99%	25%	2.5	0.04
<b>H2</b>	<b>H3+LS-GA</b>	73.73%	5.666	1.23%	4.99%	31%	10.5	0.18
<b>H2</b>	<b>H4+LS-GA</b>	73.43%	5.730	2.36%	8.88%	31%	37.9	0.63
<b>H2</b>	<b>H3+LS-KK</b>	72.93%	5.839	4.31%	15.13%	28%	1.8	0.03
<b>H1+LS-CC</b>	<b>H3+LS-RS</b>	73.88%	5.632	0.61%	5.70%	56%	50.7	0.84
<b>H1+LS-CC</b>	<b>H4+LS-RS</b>	73.58%	5.697	1.77%	7.64%	36%	48.5	0.81
<b>H1+LS-CC</b>	<b>H3+LS-GA</b>	73.95%	5.618	0.36%	4.99%	72%	247.2	4.12
<b>H1+LS-CC</b>	<b>H4+LS-GA</b>	73.78%	5.654	1.01%	5.70%	53%	514.5	8.57
<b>H1+LS-CC</b>	<b>H3+LS-KK</b>	73.48%	5.719	2.16%	9.95%	44%	42.2	0.70
<b>H2+LS-CC</b>	<b>H3+LS-RS</b>	73.88%	5.634	0.65%	5.70%	58%	51.1	0.85
<b>H2+LS-CC</b>	<b>H4+LS-RS</b>	73.60%	5.693	1.70%	7.73%	47%	50.8	0.85
<b>H2+LS-CC</b>	<b>H3+LS-GA</b>	73.91%	5.626	0.51%	4.99%	69%	335.9	5.60
<b>H2+LS-CC</b>	<b>H4+LS-GA</b>	73.76%	5.659	1.09%	7.73%	64%	515.5	8.59
<b>H2+LS-CC</b>	<b>H3+LS-KK</b>	73.44%	5.728	2.33%	10.27%	53%	41.6	0.69

Table 5.5: Summary of performance capacity cycle and day schedule generating heuristics

From Table 5.5 we see that all our heuristic combinations generate a solution within 10 minutes on average, whereas complete enumeration needs more than 8 hours on average. We observe that random search (**LS-RS**) is relatively fast and performs better than the approach of Kaandorp and Koole [29] (**LS-KK**), but it performs worse than our genetic algorithm implementation (**LS-GA**). **LS-KK** searches the solution space in a structured way and terminates when no improved schedules can be found anymore. As described in Section 4.4 we use the version of **LS-KK** with a small neighbourhood, so not all possible day schedules are generated. This might result in returning local optima, which we also observe from the relatively high average number of deferred patients and bad worst case performance when using **LS-KK**. This also explains why **LS-GA** outperforms **LS-KK**, since the randomness in both **LS-RS** and **LS-GA** makes it possible to generate all possible schedules, although by accepting only improved schedules these local search techniques can also get stuck in a local optimum. We

would expect that **LS-GA** always outperforms **LS-RS**, because the solution space of **LS-GA** is larger, so that the probability to find an improved schedule is higher than with **LS-RS**. However, when we do not use the swapping procedure for capacity cycles (**LS-CC**), random search outperforms our genetic algorithm implementation. On the other hand, when using **LS-CC** our genetic algorithm implementation outperforms random search. We observe the same in Table 5.5 when using complete enumeration for day schedule generation. This suggests that our capacity cycle heuristics have a large influence on the performance of the appointment schedules we generate, and that even complete enumeration for day schedules cannot counterbalance the performance deterioration caused by our capacity cycle heuristics.

When using heuristics instead of complete enumeration to generate day schedules, runtime often increases. This is not according to our expectations, since heuristics generate less day schedules than complete enumeration which should reduce runtime. Part of the runtime increase can be explained by the difference in technical implementation of the complete enumeration procedure and our local search techniques for day schedules. The complete enumeration implementation uses a database from which day schedules and their performance can be read, which is fast. However, our local search techniques evaluate the performance of a day schedule each time it is generated. It might thus be that schedules are evaluated multiple times, which slows down our algorithm. Another part of the runtime increase might be explained by the number of iterations needed to find a solution. With complete enumeration, in each iteration the best solution possible is selected. This results in a relatively low number of iterations needed to find an appointment schedule with balance in the number of deferred patients among days in subsequent iterations. Because **LS-RS** and **LS-GA** allocate appointments randomly to time slots, it is harder to find an appointment schedule with the same number of deferred patients in subsequent iterations. Therefore many iterations are needed to find a solution, which increases runtime. Since we can choose the solution space when we use our heuristics, we can generate appointment schedules of realistic size in reasonable time. This is not possible with complete enumeration, since the solution space would explode for large instances (see Chapter 2).

Since **LS-CC** generates additional cycles, we would expect that its performance is always better than without using this procedure. For the average number of deferred patients and the number of times that the same solution is found as with complete enumeration, this is true. However, for the worst case performance when using Heuristics 2, 3 and **LS-RS** we see that using **LS-CC** results in deteriorated worst case performance (test 233 and 582 respectively). The difference can be explained by the randomness of **LS-RS**. Only improved schedules are accepted, which makes it possible that the local search method gets stuck in a local optimum. This might be the case for the two instances as mentioned, which explains the difference.

Another remarkable observation is that our reduction heuristic (Heuristic 4) does not reduce runtime, but increases runtime on average for most combinations of heuristics. When using Heuristic 1 or 2 in combination with the swapping procedure (**LS-CC**) and random search, using our reduction heuristic results in slightly lower runtime than without the reduction heuristic. However, this difference is very small and does not outweigh the increase in the average number of deferred patients and worst case performance. Because the solution space is reduced, we expect that it is harder to find day schedules that result in balance of deferred patients among days in subsequent iterations. This results in capacity cycles with too many scheduled appointments (i.e. many deferred patients). So our reduction heuristic may result in fast execution of a single iteration, but it also causes our algorithm to run many iterations to find a solution.

We observe from Table 5.5 that the combination of Heuristic 1, 3, the swapping procedure for capacity cycles (**LS-CC**) and the genetic algorithm (**LS-GA**) for day schedules is the best performer with respect to average number of deferred patients, maximum deviation from complete enumeration and

the number of times that it obtains the same solution as complete enumeration. With this combination of heuristics appointment schedules are generated that serve on average 73.95% of the walk-in patients on the day they arrive. Compared to the 74.04% of the walk-in patients that are served on the day of their arrival as found with complete enumeration, we can conclude that the combination of Heuristics 1, 3, **LS-CC** and **LS-GA** comes very close to the result obtained with complete enumeration. Its worst case performance is the same as when using complete enumeration for day schedules. In that setting we use arrival rate pattern 2 (Figure 5.2), walk-in patients are willing to wait a maximum of 2 time slots, the no-show percentage is 15% and 95% of the patients with an appointment request need to be served within 5 days from their request. To show where this best performing combination of heuristics deviates from complete enumeration, we present in Figure 5.5 the appointment schedules created by complete enumeration (light coloured bars) and the combination of Heuristics 1, 3 **LS-CC** and **LS-GA** (dark coloured bars) compared to the walk-in arrival rates (line).

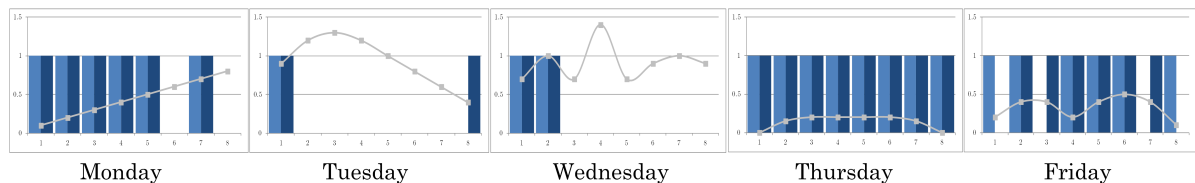


Figure 5.5: CAS with **CE+CE** and **H1+H3+LS-CC+LS-GA** vs. walk-in arrival rates

We observe from Figure 5.5 that on Monday the best performing combination of heuristics gives the same appointment schedule as complete enumeration. Since the walk-in arrival rate is increasing over the day and walk-in patients are willing to wait 2 time slots, leaving time slots open at the end of the day is beneficial. We observe that we do not schedule an appointment in the last time slot, because a relatively high number of expected walk-in patients would then be deferred. On Tuesday we see that complete enumeration only schedules one appointment, whereas the best performing combination of heuristics schedules two appointments. Again, walk-in patients are willing to wait for service so planning an appointment in the first time slot does not directly lead to deferral of many walk-in patients. However, our heuristics schedule also an appointment in the last time slot. This results in a high probability that the walk-in patients that arrive in that time slot get an appointment on another day. This seems counterintuitive, but it might be that the relatively low walk-in rate in this time slot causes our heuristics to schedule an appointment in the last time slot. On Wednesday we observe again that the best performing combination of heuristics gives the same schedule as complete enumeration. Again the waiting behaviour of walk-in patients is exploited by scheduling appointments early on the day. On Thursday the walk-in arrival rate is very low over the day. Therefore the entire day is used to serve patients with an appointment. Again the best performing combination of heuristics gives the same schedule as complete enumeration. The combination of heuristics gives a different schedule than complete enumeration on Friday, with an equal number of scheduled appointments as complete enumeration. However, the performance with respect to the number of deferred patients is the same for both schedules. This implies that there is not always a single best performing day schedule, given a certain number of appointments to schedule. We can conclude that the performance difference in this worst case scenario is caused by the Tuesday schedule, since complete enumeration needs one appointment less than the combination of heuristics on that day.

When we compare the average results in Table 5.5 with Table 5.4 where we use complete enumeration to generate day schedules, we see that our GA implementation results in the same performance as when complete enumeration is used to generate day schedules. This suggests again that the deviation from complete enumeration is caused by our capacity cycle generating heuristics, when using Heuristics 1,

3, **LS-CC** and **LS-GA**. Observe as well that choosing Heuristic 1 instead of Heuristic 2 results in only a slightly lower fraction of walk-in patients served on the day of their arrival. Since the performance differences with respect to deferred patients are small, we select the three best performing combinations to test in our case study in Chapter 6. The three best performing combinations of heuristics are Heuristics 1, 3, **LS-CC** and **LS-GA**, Heuristics 2, 3, **LS-CC** and **LS-GA**, Heuristics 1, 3, **LS-CC** and **LS-RS**.

## 5.4 Summary & conclusion

We tested all combinations of heuristics and local search techniques on 36 problem instances and we can conclude that most of the combinations of heuristics are able to come up with a feasible solution within one minute of runtime, which is much faster than complete enumeration. Because of randomness in the local search techniques and differences in technical implementation, our day schedule heuristics might result in a higher runtime than using complete enumeration to generate day schedules for small instances. This would support the use of complete enumeration for day schedules. However, for large instances heuristics are the only option to generate an appointment schedule, since complete enumeration takes too many computations.

The fraction of walk-in patients served on the day of their arrival is within 2 percent point from complete enumeration. The combination of Heuristics 1, 3, **LS-CC** and **LS-GA** is the overall best performer when not taking runtime into account. With this combination of heuristics we obtained in our tests the same performance with respect to the number of deferred patients as when using complete enumeration to generate day schedules. The worst case performance of this combination of heuristics is obtained when the access time service level norm is set on serving 95% of the appointment requests within 5 days. This is observed for all tested patterns. This implies that relatively many patients get deferred in case this combination of heuristics is applied in a facility that sets a low access time norm. We observe the same worse case behaviour when using complete enumeration for day schedule generation, which suggests that our capacity cycle heuristics are the cause of this deviation. We would therefore advise to conduct further research in improving capacity cycle generating heuristics.

Since we only test on relatively small instances, we cannot be completely sure whether the best performing combination of heuristics in small tests is also the best performer on large instances. Therefore we select the three best performing combinations of heuristics to assess in our case study in Chapter 6. These are the combinations of Heuristics 1, 3, **LS-CC** and **LS-GA**, Heuristics 2, 3, **LS-CC** and **LS-GA**, and Heuristics 1, 3, **LS-CC** and **LS-RS**.

# Chapter 6

## Case study

In this chapter we assess the performance of our algorithm and that of benchmarks from literature on data of the CT-scan facility of the AMC. In Section 6.1 we describe how a CT-scan is made. In Section 6.2 we discuss the setup of our algorithm in our case study. In Section 6.3 we present the results of the case study. We end this chapter with a summary and conclusions in Section 6.4.

### 6.1 Making a CT-scan

The radiology department of the AMC uses medical imaging technologies for diagnostic examinations. Each day about 400 to 500 patients visit the radiology department and each year more than 200,000 diagnostic examinations are performed [2]. Available imaging technologies are regular x-ray technology, ultrasound technology, Magnetic Resonance Imaging (MRI) and Computed Tomography (CT). The images created by these technologies are used by medical specialists to support the right treatment of the patient. The CT-scan facility is part of the radiology department and each year more than 10,000 patients undergo a CT-scan in the AMC [38]. In this case study we use data of that CT-scan facility.

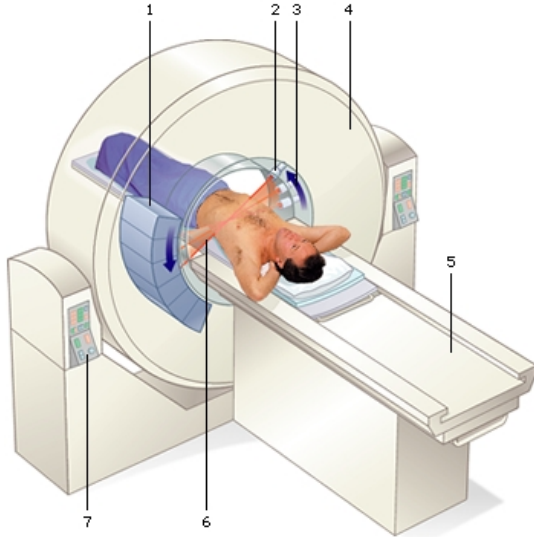
A CT-scanner makes an image of (a part of) the body. An image consists of thin slices (with thickness varying from 1 to 10 mm.), that are made with the use of x-rays. In the final image a distinction can be made between tissues with different densities. This makes it possible to distinguish bones and blood vessels for example. In Figure 6.1a we present a schematic overview of a diagnostic examination with a CT-scanner and in Figure 6.1b we present examples of images created with a CT-scanner.

In Figure 6.1a, (1) indicates the x-ray detector. This detector turns around, just like the x-ray source (2). The detector and source turn in a fixed direction (3). For each slice of the image, one rotation is made. In Figure 6.1a, (4) indicates the housing of the CT-scanner. The scanner can be set on an angle, to make an image of the body from another direction. When a CT-scan is being made, the patient lies on a bed (5) that moves through the scanner. Number (6) indicates the x-ray beam and (7) indicates a manual control panel. Usually, the laboratory worker who makes the CT-scan controls the scanner from another room such that this person is protected against the x-rays in the scanning room.

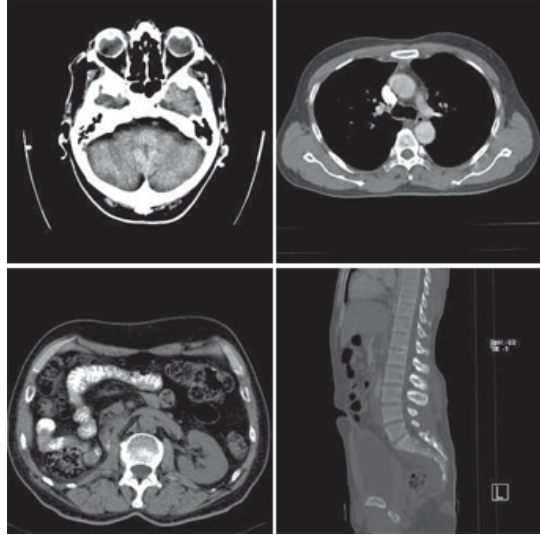
### 6.2 Case study setup

In this section we discuss the setup of our algorithm, that we use in our case study. To evaluate large instances, we modified the method that Kortbeek et al. [37] use to evaluate the performance of day schedules. We discuss these modifications in Section 6.2.1. In Section 6.2.2 we present the data set





(a) Schematic overview CT-scanner [30]



(b) Example of images created by a CT-scanner [1]

that we use in the case study. We compare the performance of our algorithm with benchmarks from literature. In Section 6.2.3 we describe these benchmarks.

### 6.2.1 Technical modifications

In our first tests on large instances, it appeared that the algorithm of Kortbeek et al. [37] gave unreliable results. Namely, a negative number of walk-in patients got deferred, which is of course not possible. After we evaluated the code of the software they wrote around their algorithm, we concluded that the negative number of deferred patients is caused by the generating function approach that Kortbeek et al. [37] use to evaluate day schedules. They use an approach based on generating functions to derive expressions for the distribution of the backlog at the start of each day in the planning cycle. Therefore we had to modify this approach to find reliable results for the backlog distribution at the start of each day in the cycle.

Kortbeek et al. [37] give the transition probabilities for going from a day  $d$  with a backlog that equals  $i$ , to day  $d + 1$  with a backlog that equals  $i'$ . From these transition probabilities, the stationary probabilities that at the start of day  $d$ , the backlog equals  $j$  jobs can be found. Instead of using a generating function approach, we solve the corresponding linear balance equations with Gauss Jordan elimination. To let our approach work, we have to decide on a value for the maximum expected backlog on a day. We use a maximum backlog of 100. From our experiments it appears that the probability that this backlog is realised on a certain day is generally lower than  $10^{-3}$ , which supports our choice. We implemented our approach with the Embarcadero Delphi XE programming package.

### 6.2.2 Settings CT-scan facility

For the case study we use data that was collected in 2008 for an earlier study in the AMC [38]. Although this data might be outdated, we can still use it to assess the performance of our algorithm. The purpose of this case study is to show that our algorithm performs better than appointment rules from literature. It should be noted that when applying our algorithm in practice at the CT-scan department of the AMC, a new data analysis should be done to obtain reliable and up-to-date estimates of the input parameters. Up-to-date information is necessary, otherwise our algorithm would probably give unsatisfactory appointment schedules. Other facility characteristics we use in this case study, such as

the number of available scanners, are also based on the situation in 2008.

In 2008, the AMC had three CT-scanners available. However, one of them was dedicated for scanning emergency patients such that we take it out of our analysis. Since part of the patients (15.4% [38]) needs two time slots for service, we adjust the walk-in arrival rates and appointment request rates for these patients. One of the assumptions of our algorithm is that service times are deterministic and equal to one time slot, so we cannot schedule appointments with a duration of two time slots (equal to 30 minutes). To incorporate the effect of these longer appointments on the workload of the CT-scan facility, we adjust the walk-in arrival rates and appointment request rates. Therefore we multiply the walk-in arrival rates for each time slot with  $0.154 \cdot 2 + 0.846 \cdot 1 = 1.154$ . The workload of the facility under study becomes then 62.3%.

Each day the facility is 8.5 hours operational (facility opens at 8:00 and closes at 16:30), such that the day schedules in this case study consist of 34 time slots of 15 minutes. The appointment schedule we create has a length of 5 days, such that the resulting appointment schedule can be repeated each week. The time that walk-in patients are willing to wait is set on 2 time slots, which is equal to 30 minutes. In earlier research at the AMC towards patient preferences it was found that 80% of the walk-in patients want to be served within 30 minutes [52]. The same research concludes that patients with an appointment request prefer to be served within 11 days. The current access time norm of the AMC is 10 days [14], so we set the access time service level norm for our case study such that 95% of the patients with an appointment request should be served within 10 days. Figures about no-show rates differ a lot in literature. However, figures between 10% and 20% are most common [11, 28, 58]. In an earlier study at the AMC towards the feasibility of walk-in at the CT-scan facility, a no-show rate of 3.1% was found [38]. Because this is low compared to literature and this number is only based on registered no-shows (so it might be that part of the no-shows were not registered), we set the no-show rate for our case study at 5.0%. We give an overview of the CT-scan facility parameters in Table 6.1.

Parameter	Description	Parameter value
$R$	Number of available resources	2
$D$	Length of the planning cycle	5 days
$h$	Length of a time slot	15 minutes
$T$	Total number of available time slots per resource per day	34
$g$	Maximum number of time slots that a walk-in patient accepts to wait for service	2 (i.e. 30 minutes)
$q$	No-show probability	0.05
$(y, S^{norm}(y))$	Access time service level norm: fraction of jobs with access time not greater than $y$ is at least $S^{norm}(y)$	(10, 95%)

Table 6.1: An overview of the CT-scan facility characteristics used in our case study

For the appointment request rate and walk-in arrival rates we use estimated values from earlier research conducted on 2008 data [38, 52]. It appears from these studies that from all the patients that need a CT-scan, 76.4% prefers walk-in. This means that 23.6% of all patients prefers to make an appointment for a visit to the CT-scan. In those studies an arrival pattern is presented for the case that all patients are walk-in patients. This pattern shows arriving walk-in patients from 7:00 to 17:45, so also outside opening hours. We treat the walk-in patients that arrive outside opening hours according to that pattern as patients with an appointment request, since they cannot be served on the day they arrive. A part of the remaining walk-in arrivals (so the expected rates during opening hours) also prefers an appointment. This part is found by multiplying the arrival rate of walk-in patients per time slot with the probability that a patient prefers an appointment and sum these values per day. To generate the

walk-in arrival rate pattern we multiply the walk-in arrival rate per time slot and per day with the percentage of patients that prefers walk-in. We display the resulting expected walk-in arrival rates and appointment request rates in Figures 6.1 and 6.2 respectively.

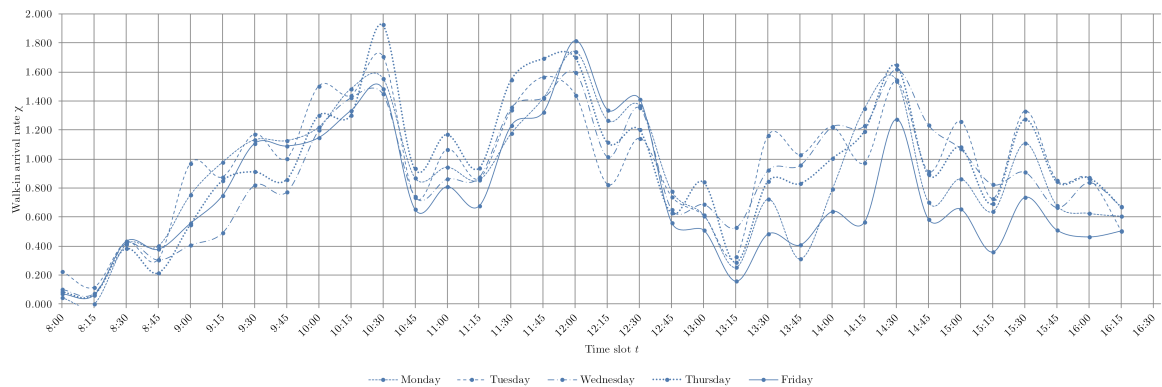


Figure 6.1: Walk-in arrival rates AMC CT-scan facility [38]

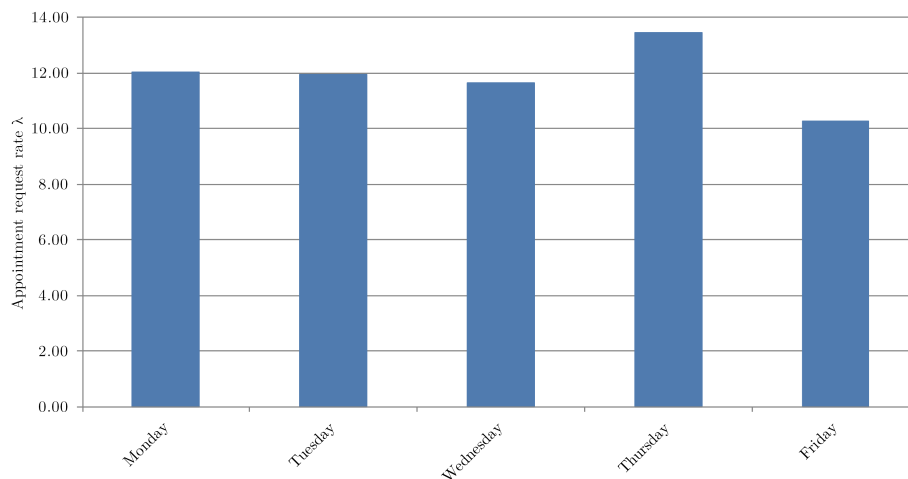


Figure 6.2: Appointment request rates AMC CT-scan facility [38, 52]

### 6.2.3 Benchmarks

In our literature review we identified appointment scheduling rules that appeared to be effective in generating day schedules. We use three different benchmarks. In our first benchmark (**B1**) we schedule appointments in alternating sequence over the day. Su and Shih [56] describe this approach, and a similar approach is described by Klassen and Rohleder [33], who state that urgent slots (walk-in patients in our case) should be spread evenly over the day. In our second benchmark (**B2**) we first schedule a block of time slots where only walk-in patients can be served, then we schedule a block of appointments and finally we schedule a block for walk-in patients again. We assume that the blocks for walk-in patients are of equal size. Chen and Robinson [15] present this approach and show that it performs well in generating day schedules. Our third benchmark (**B3**) is a combination of the previous two benchmarks. We schedule blocks of appointments and blocks where arriving walk-in patients can be served in alternating sequence over the day. We choose the length of these blocks equal to the time  $g$  that walk-in patients are willing to wait, so that walk-in patients that arrive in the first time slot of an appointment block can be served after this block. We give an example of the three benchmarks

in Figure 6.3, where black squares indicate slots reserved for appointments and white squares indicate slots that are free to serve walk-in patients.

In our literature review (Chapter 3) we concluded that methods to determine how many appointments to schedule per day in a planning cycle are scarce. Therefore we decide to generate capacity cycles for our benchmarks in the same way as we do for our own algorithm. We run our algorithm with the capacity cycle generating heuristics 1 and 2 and the swapping procedure for capacity cycles **LS-CC**, while using the benchmarks to generate day schedules. Per benchmark we select the best performing schedule (performance may be different when using different combinations of capacity cycle generating heuristics). We also use the iterative character of our algorithm. In practice we would expect that planners adjust their schedules if it appears that too many walk-in patients are deferred, waiting time for walk-in patients is too high and when the access time service level norm for patients with an appointment is violated. This justifies the use of the iterations in the benchmarks, because these iterations have the same adjusting effect. This approach may lead to better appointment schedules than a planner would make (manually) in practice (since many options are evaluated and we use iterations). However, when our algorithm still outperforms the benchmarks, this supports the applicability of our algorithm in practice. To the best of our knowledge this is the fairest approach to compare the performance of our algorithm with that of benchmarks.

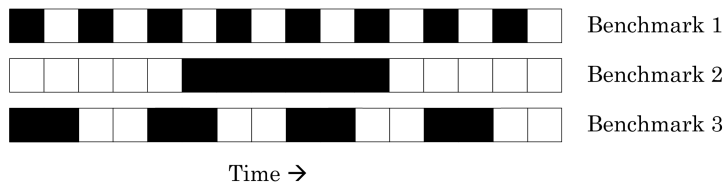


Figure 6.3: Benchmarks day schedule generation

## 6.3 Results

In this section we present the results with respect to the number of deferred patients and runtime for the three best performing combinations of heuristics that we identified in Chapter 5. We also present the performance of the three benchmarks from literature. When applying the best performing algorithms from Chapter 5 on the case study test instance, it appeared that no solution could be found when using the local search parameter settings (e.g. number of neighbour schedules for random search) as used for small instances in Chapter 5. Therefore we increased the number of neighbour schedules to generate for random search and the population size for the genetic algorithm. In Chapter 5 we found appointment schedules with high performance when using 10 neighbour schedules with random search and when using a population size of 10 with the genetic algorithm implementation. First we increased these values to 30 and 50, but again no solution could be found within 48 hours runtime. This is probably due to the randomness in both local search methods, and the highly increased solution spaced of our case study test instance compared to the small test instance in Chapter 5. Finally we found a solution when using 75 neighbour schedules for random search and a population size of 75 for the genetic algorithm implementation. We give a summary of these results in Table 6.2.

From Table 6.2 we observe that all combinations of heuristics we tested perform significantly better with respect to the number of deferred patients than the benchmarks. We tested the benchmarks with the four possible capacity cycle generating heuristics (**H1**, **H2**, **H1+LS-CC** and **H2+LS-CC**). It appears that all benchmarks show the best result with respect to the number of deferred patients when we use Heuristic 1 in combination with the swapping procedure for capacity cycles **LS-CC**.

Capacity cycle method	Day schedule method	Fraction of walk-in patients served on day of arrival	Number of deferred patients	Runtime (minutes)	Runtime (hours)
<b>H1+LS-CC</b>	<b>H3+LS-GA</b>	99.42%	0.890	1360.5	22.68
<b>H2+LS-CC</b>	<b>H3+LS-GA</b>	99.43%	0.875	1102.1	18.37
<b>H1+LS-CC</b>	<b>H3+LS-RS</b>	99.43%	0.869	82.9	1.38
<b>H1+LS-CC</b>	<b>Benchmark 1</b>	97.58%	3.687	5.1	0.09
<b>H1+LS-CC</b>	<b>Benchmark 2</b>	91.98%	12.221	7.6	0.13
<b>H1+LS-CC</b>	<b>Benchmark 3</b>	97.67%	3.553	5.0	0.08

Table 6.2: Overview of performance benchmarks and our algorithm

The best performing combination of heuristics is the combination of Heuristics 1, 3, **LS-CC** and **LS-RS**. This combination serves 99.43% of the walk-in patients on the day of their arrival. The best performing benchmark with respect to the number of deferred patients is Benchmark 3. However, the best performing combination of heuristics defers 75.5% less patients than this best performing benchmark. This is equal to almost a 2 percent point increase in the fraction of walk-in patients served on the day of their arrival. These promising results might be explained by the relatively low workload of the system under study. We expect that under a higher workload than the 62.3% in our case study, the fraction of walk-in patients served on the day of their arrival decreases.

Under a high workload we expect to have less flexibility in allocating appointments, which may increase the number of deferred patients. To get a first idea about the influence of a high workload, which is not observed in the 2008 data set but that might occur in the future, we scale the data set from the case study up such that the workload becomes 85%. For this data set, we generate an appointment schedule with our best performing combination of heuristics (i.e. H1, 3, 1, 3, **LS-CC** and **LS-RS**) and the three benchmarks. Under this relatively high workload, we observe that the fraction of walk-in patients served on the day of their arrival remains high on 94.16%. The best performing benchmark under this high workload is again Benchmark 3 in combination with Heuristic 2 to generate capacity cycles and it generates a schedule where 86.78% of the walk-in patients can be served on the day of their arrival. Although more extensive testing under a high workload seems necessary, this first test shows that the difference between our algorithm and the best performing benchmark increases. Furthermore, the combination of Heuristics 1, 3, **LS-CC** and **LS-RS** generates a schedule under the high workload in 3.2 hours. This suggests that runtime will increase under a high workload, but again, more tests are necessary to confirm this hypothesis. In Appendix F we give a summary of these results.

The capacity cycle found in the last iteration of the best performing combination of heuristics (i.e. Heuristics 1, 3, **LS-CC** and **LS-RS**) is equal to (14,10,10,10,17) (details can be found in Appendix E). This result confirms the findings of Sickinger and Kolisch [53], who state that a simple random search procedure outperforms more advanced local search methods. The difference with the second and third best combination of heuristics is small, however these combinations have a much higher runtime than the best performing combination. To achieve the access time service level norm it is required to schedule 61 appointments, while the number of appointment requests in the last iteration is 60.205. This means that only a buffer capacity of 0.795 is needed to account for the variability in appointment request arrivals, so capacity is used efficiently.

Now we discuss the day schedules generated in the last iteration by the combination of Heuristics 1, 3, **LS-CC** and **LS-RS**, which is the best performing combination. Since walk-in patients are willing to wait 30 minutes (i.e. 2 time slots) for service, it is beneficial to use all available resources for appointments in the first two time slots. Arriving walk-in patients can then be served in the third time slot. This is what we observe from the day schedule for Monday, as we present in Figure 6.4. The relatively low walk-in arrival rate at the start of the day causes our algorithm to schedule appointments at the start of the day. Around 9:30 hours the walk-in arrival rate becomes that high, that the algorithm leaves all resources free to serve arriving walk-in patients. Just after lunch (around 13:15 hours) relatively few walk-in patients are expected to arrive, which leads to reserving time slots for patients with an appointment. The same holds for the end of the day, with a low walk-in arrival rate. Observe that after the morning never more than 2 time slots after each other are reserved for appointments. This is to make sure that arriving walk-in patients can be served within  $g$  time slots, which is equal to 30 minutes in our case study.

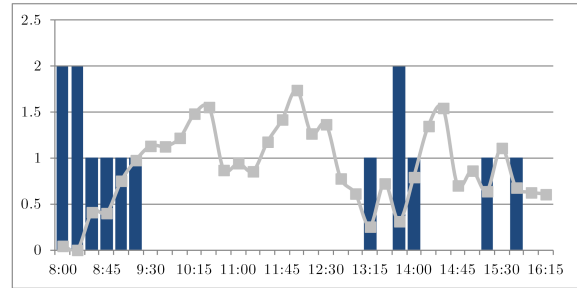
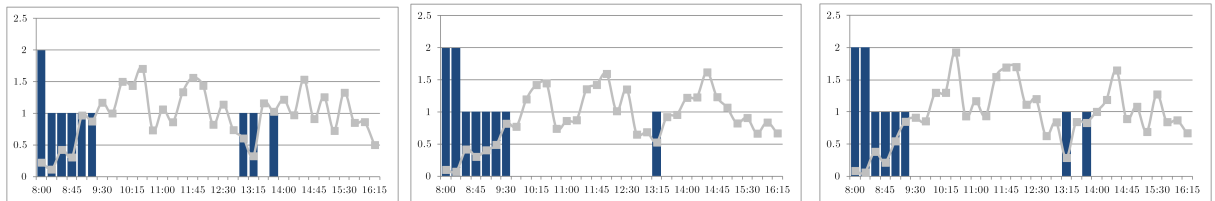


Figure 6.4: Appointment schedule and walk-in arrival rate on Monday



(a) Appointment schedule and walk-in arrival rate on Tuesday (b) Appointment schedule and walk-in arrival rate on Wednesday (c) Appointment schedule and walk-in arrival rate on Thursday

Figure 6.5: Appointment schedule and walk-in arrival rates

From Tuesday to Thursday similar day schedules are generated, as we observe from Figure 6.5 a, b and c. This is according to our expectations, since the walk-in arrival rate patterns do not differ that much among those days and the number of appointments to plan is the same for Tuesday, Wednesday and Thursday. Like on Monday, appointments are scheduled at the start of the day where the walk-in arrival rate is relatively low. We see that our algorithm also schedules appointments around lunch, since the walk-in arrival rate is low at that moment of the day. Again, no more than 2 time slots are used after each other to serve patients with an appointment. This is logical, since the allowed waiting time for walk-in patients is then not violated.

On Friday relatively many appointments have to be scheduled. Again we observe that those appointments are scheduled around peaks in walk-in arrival demand. A difference with the previous days is that on Friday more than 2 appointments are scheduled after each other at the end of the day. However, the CT-scan facility has 2 available resources. By reserving one CT-scanner for walk-in patients in the afternoon, arriving walk-in patients can be served with that scanner. Since

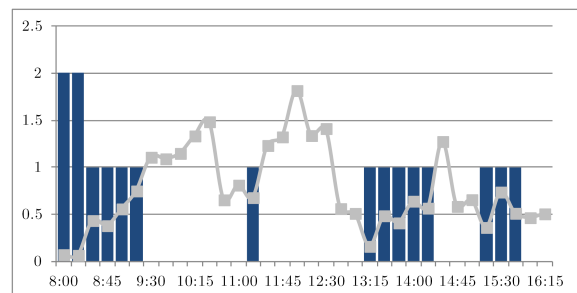


Figure 6.6: Appointment schedule and walk-in arrival rate on Friday

the walk-in arrival rate is low during the blocks of scheduled appointments in the afternoon, 1 scanner should be sufficient to serve the arriving walk-in patients.

All day schedules in our case study show that it is beneficial to schedule appointments in time slots where few walk-in patients are expected to arrive, so around peaks with high expected walk-in arrival rates. We also observe this for the test we performed on a instance with a workload of 85%, as we present in Appendix F. This is according to our expectations, since this balances the workload over a day.

## 6.4 Summary & conclusion

In this chapter we presented the results of our case study. We conduct our case study with data of the CT-scan facility at the AMC gathered in 2008 for an earlier study. We gave a short overview of how a CT-scan is being made and which characteristics the CT-scan facility of the AMC has. Based on data from this facility we compare the three best performing combinations of heuristics from Chapter 5 with benchmarks from literature. It appears that for the situation as it was in 2008 our best performing algorithm (i.e. Heuristics 1, 3, **LS-CC** and **LS-RS**) is able to find an appointment schedule in less than 1.5 hours and serves 99.43% of the walk-in patients on the day they arrive. This result confirms earlier research that states that a random search implementation can outperform more advanced local search methods.

The benchmarks do not take the walk-in arrival pattern into account, but they generate a fixed schedule with some preset appointment rule. This results in many deferred patients, because the benchmarks also schedule appointments in time slots where we expect relatively many walk-in patients to arrive. Our algorithm exchanges appointments in time slots randomly during the search for better day schedules (both for the genetic algorithm implementation and random search), such that no appointments are scheduled in time slots with many expected walk-in arrivals. In our case study we show that our algorithm balances workload by planning appointments in time slots with few expected walk-in arrivals, while leaving time slots open at moments where we expect many walk-in patients to arrive. This results in 75.5% less deferred patients than found with the best performing benchmark from literature.

# Chapter 7

## Conclusions and recommendations

In this chapter we present the conclusions of our research. In Section 7.1 we present our conclusions. In Section 7.2 we touch on elements of our research that might be subject to discussion. In Section 7.3 we end this chapter with interesting directions for further research.

### 7.1 Conclusions

Traditionally diagnostic facilities schedule appointments for all patients that require an examination. Allowing patients to walk in without an appointment reduces access times. It also creates the possibility to combine outpatient consultations and diagnostic examinations on one day, which speeds up the diagnostic process. Since not all patients can walk in, our algorithm generates appointment schedules with which both patients with an appointments and walk-in patients can be served. The generated schedule prescribes the number of appointments to plan per day and the moment on the day to plan these appointments. We maximize the fraction of walk-in patients that can be served on the day of their arrival (which is equal to minimising the number of deferred patients), while satisfying an access time service level norm for patients with an appointment.

The best performance with respect to the number of deferred patients and runtime is obtained when using a capacity cycle generating heuristic based on the mean number of free time slots per day (**H1**) in combination with the swapping procedure to generate additional cycles (**LS-CC**), a day schedule generating heuristic that generates an initial day schedule based on mean walk-in arrival rates (**H3**) and a random search technique (**LS-RS**) to find improved day schedules. We observe that deviations from the solutions found for problem instances with the algorithm of Kortbeek et al. [37] (i.e. by complete enumeration) were caused by the capacity cycle generating heuristics. It appears that the day schedule generating heuristics are able to come up with the same schedules as found with complete enumeration. Apparently it is harder to determine the total number of appointments to plan in a cycle and to allocate these appointments over the days in the cycle, than to allocate appointments to time slots of a day.

When we apply our algorithm to data from the CT-scan facility of the AMC, we see that our best performing algorithm (i.e. **H1+H3+LS-CC+LS-RS**) is able to defer 75.5% less patients than the best performing benchmark from literature. This combination is also the fastest combination of heuristics that we assessed in the case study. With a workload of 62.3% (case study) it serves 95% of the patients with an appointment request within 10 days and 99.43% of the walk-in patients is served on the day of their arrival. This means that for a practical situation, as in the case study, almost all walk-in patients get consultation and diagnostic examination at the same day (one-stop-shop). The appointment schedule from the case study is generated within 1.5 hours. This is even faster than when



using complete enumeration for small instances, so our algorithm is much more efficient in generating appointment schedules than complete enumeration with respect to runtime. It might even be faster than generating an appointment schedule manually, which also supports the applicability of our algorithm in practice. In a test on a scaled-up case study instance with a workload of 85%, this algorithm serves 94.16% of the walk-in patients on the day of their arrival and it defers 55.8% less patients than the best performing benchmark. The corresponding appointment schedule was generated in 3.2 hours. These results are promising, however more extensive testing is necessary to confirm these findings.

Our algorithm seems to be sensitive to a tight access time service level norm. The tightest access time norm we tested for was equal to serving 95% of the appointment requests within 5 days and for this instance our algorithm gave 5% deviation in the number of deferred patients compared to complete enumeration on small instances. At the moment, the preferred maximal access time for CT-scans in the AMC is 10 days, under which conditions our algorithm performs good (both on small instances and in the case study). In all cases we evaluated we see that our algorithm performs better than the benchmarks from literature we tested for. Although the case study shows promising results with respect to the performance of our algorithm, we doubt direct applicability in practice. To clarify this point, we discuss limitations of our research in Section 7.2 and suggest interesting directions for further research in Section 7.3.

## 7.2 Discussion

Our research is based on the approach described by Kortbeek et al. [37]. The assumptions in our model are therefore equal to the assumptions Kortbeek et al. [37] make in their model. We think that some assumptions might not completely reflect reality, which might be an issue in implementing our algorithm in practice. The first assumption is that service times in the model are assumed to be deterministic and of equal length. This is a reasonable approximation for a CT-scan department where almost all appointments have the same duration, but this might be different for other applications with more variability in service times. Examples where our model could be applied but where service times are probably non-deterministic are car repair shops or service engineers that serve requests on appointment, but that also have to deal with emergency requests. Examples with scheduled appointments and (urgent) walk-in with variable service times are MRI-scan facilities and operating theatres.

In our model patients of only one type can be served. However, in practice many patient types (or customer types) should be served that all have different characteristics. Characteristics that may differ among patients are for example the time they are willing to wait for service and their preparation time before they can receive service (e.g. some patients need to take contrast fluid before a CT-scan).

In our tests to find the best performing combination of heuristics and local search techniques we only tested on small instances (i.e. 1 resource, a planning cycle of 5 days and 8 time slots per day), to compare the performance of our heuristics with complete enumeration. However, it might be that the best performing heuristics on small instances are not the best performing heuristics for larger instances. This is what we observe in the case study in Chapter 6. On the other hand, we show that the best performing heuristics from our tests on small instances are all able to generate appointment schedules of good quality in the case study. These schedules are generated fast and a large part of the walk-in patients can be served on the day of their arrival, which is the aim of our research.

A last point of discussion is the importance of our algorithm's runtime. Since a planner would like to generate schedules as fast as possible, a very low runtime would be preferable. However, our algorithm can be applied on a tactical level. This means that our algorithm generates schedules for a long term (e.g. a season of three months), which implies that our algorithm should generate a schedule only a few times per year. In that case, runtime is less important. We think that a trade-off should

be made between high algorithm performance (high runtime) and usability in practice (relatively low runtime, to reduce the risk of losing data in case of incidents (e.g. power failures)).

### 7.3 Suggestions for further research

In our study we assess the performance of our algorithm on a problem instance with input parameters from practice. As discussed in Section 7.2, practice is not fully reflected in our model. We think it is of interest to assess the performance of the appointment schedules generated with our algorithm in a simulation study. Simulation studies are very flexible and can be used to make a detailed reflection of practice. We think it is also important to actively collaborate with the CT-scan (or other demanding) facility at the AMC, such that commitment and trust in the new appointment scheduling algorithm are established. In case the simulation study shows that the assumptions of our model are too restrictive for an application in practice, modelling of stochastic service times or different patient types might be necessary.

Before our algorithm can be applied in practice, it should be clear how often new appointment schedules have to be generated. Since our algorithm generates cyclic schedules on the tactical planning level, it is not necessary to update the schedules each week. However, arrival rates of walk-in patients and appointment requests may differ among periods in the year. Therefore we suggest to perform a data analysis on historical data to make clusters of periods with equal characteristics, for which appointment schedules can be made.

In our model we assume that part of all patients with an appointment does not show up (i.e. a no-show). To effectively use resources, we think it is interesting to study the possibility to overbook some time slots. By allowing more demand than capacity in the facility, the negative effect of no-shows can be neutralised. It would be interesting to study which time slots should then be overbooked.

Another suggestion for further research is the optimisation of the local search techniques we use. As we described in Section 5.2.2, we configure our local search techniques only once. We generate for example 10 neighbour schedules with our random search technique in each iteration of our algorithm, although a lower or higher number of neighbour schedules might also be possible. This might influence the test results. Fortunately, we show that the choices we make result in good performing heuristics compared to complete enumeration. It might however be that still too many capacity cycles and day schedules are evaluated, which slows down our algorithm but results in good performance. Therefore we propose to study the effect of local search neighbourhood size on algorithm performance with respect to deferred patients and runtime.

A last interesting direction for further research is developing a better capacity cycle heuristic or adapting our local search technique for capacity cycles, since a large part of the deviation from complete enumeration is caused by our capacity cycle heuristics. Our algorithm generates neighbouring capacity cycles based on an initial cycle in one step and applies a day schedule generating technique on all of these cycles. Finally we choose the best performing combination of capacity cycle and corresponding day schedules. However, generating capacity cycles step-by-step might give better results. Our approach would then be to generate an initial capacity cycle and apply day schedule generating techniques on this cycle. The next step would be to modify the initial capacity cycle (e.g. at random) and apply day schedule generating techniques again. If this modified capacity cycle has improved performance compared to the initial cycle, we accept this new cycle and otherwise we keep working with the initial cycle. We repeat this procedure until a stopping criterion is met (e.g. a fixed number of capacity cycles are generated). We expect that this approach gives good results, since in each step a better solution can be found. This is not true for our current approach. We would also advise to include a way to escape from local optima, by for example accepting worse performing schedules with certain probability.

# Bibliography

- [1] AMC Amsterdam. Een CT-scan, wat is dat? Downloaded on 28 June 2012, from <http://www.amc.nl/web/Zorg/Patient/Patienteninformatie/CTscan.htm>, n.d.
- [2] AMC Amsterdam. The Radiology Department. Downloaded on 28 June 2012, from <http://www.amc.nl/web/Het-AMC/Afdelingen/Medische-afdelingen/Radiologie/Radiologie/Afdeling.htm>, n.d.
- [3] R. Ashton, L. Hague, M. Brandreth, D. Worthington, and S. Cropper. A Simulation-Based Study of a NHS Walk-in Centre. *The Journal of the Operational Research Society*, 56(2):153 – 161, 2005.
- [4] H. Balasubramanian, A. Muriel, and L. Wang. The impact of provider flexibility and capacity allocation on the performance of primary care practices. *Flexible Services and Manufacturing Journal*, Online:1 – 26, 2011.
- [5] R.A. Berenson, T. Bodenheimer, and H.H. Pham. Specialty-Service Lines: Salvos In The New Medical Arms Race. *Health Affairs*, 25(5):337 – 343, 2006.
- [6] D. Bertsimas and R. Shioda. Restaurant Revenue Management. *Operations Research*, 51(3):472 – 486, 2003.
- [7] L. Bianchi, M. Dorigo, Gambardella L.M., and W.J. Gutjahr. A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing*, 8:239 – 287, 2009.
- [8] G.R. Bitran and S.M. Gilbert. Managing Hotel Reservations with Uncertain Arrivals. *Operations Research*, 44(1):35 – 49, 1996.
- [9] A.P. Carpenter, L.M. Leemis, A.S. Papir, D.J. Phillips, and G.S. Phillips. Managing magnetic resonance imaging machines: support tools for scheduling and planning. *Health Care Management Science*, 14(2):158 – 173, 2011.
- [10] T. Cayirli and E. Veral. Outpatient scheduling in health care: a review of literature. *Production and Operations Management*, 12(4):519 – 549, 2003.
- [11] T. Cayirli, E. Veral, and H. Rosen. Designing appointment scheduling systems for ambulatory care services. *Health Care Management Science*, 9:47 – 58, 2006.
- [12] T. Cayirli, E. Veral, and H. Rosen. Assessment of Patient Classification in Appointment System Design. *Production and Operations Management*, 17(3):338 – 353, 2008.
- [13] T. Cayirli, K.K. Yang, and S.A. Quek. A Universal Appointment Rule in the Presence of No-Shows and Walk-Ins. *Production and Operations Management*, 21(4):682 – 697, 2012. 10.1111/j.1937-5956.2011.01297.x.
- [14] Academisch Medisch Centrum. AMC Patiëntenmanifest. Amsterdam, May 2011.

- [15] R.R. Chen and L.W. Robinson. Sequencing and Scheduling Appointments with Potential Call-In Patients. UC Davis Graduate School of Management Research Paper, June 2011.
- [16] S. Creemers and M. Lambrecht. An advanced queueing model to analyze appointment-driven service systems. *Computers & Operations Research*, 36(10):2773 – 2785, 2009.
- [17] S. Creemers and M. Lambrecht. Queueing models for appointment-driven systems. *Annals of Operations Research*, 178(1):155 – 172, 2010.
- [18] B. Denton, A.S. Rahman, H. Nelson, and A.C. Bailey. Simulation of a multiple operating room surgical suite. In *Proceedings of the 2006 Winter Simulation Conference*, 2006.
- [19] K.F. Doerner, W.J. Gutjahr, R.F. Hartl, C. Strauss, and C. Stummer. Pareto Ant Colony Optimization with ILP preprocessing in multiobjective project portfolio selection. *European Journal of Operational Research*, 171:830 – 841, 2006.
- [20] D. Falsini, A. Perugia, and M.M. Schiraldi. An Operations Management Approach for Radiology Services. In *Proceedings of the Conference on "Sustainable Development: Industrial Practice, Education & Research"*. Bari, Italy, September 2010.
- [21] A. Freville and G. Plateau. An efficient preprocessing procedure for the multidimensional 0-1 knapsack problem. *Discrete Applied Mathematics*, 49:189 – 212, 1994.
- [22] Y. Gocgun, B.W. Bresnahan, A. Ghate, and M.L. Gunn. A Markov decision process approach to multi-category patient scheduling in a diagnostic facility. *Artificial Intelligence in Medicine*, 53:73 – 81, 2011.
- [23] L.V. Green, S.V. Savin, and B. Wang. Managing Patient Service in a Diagnostic Medical Facility. *Operations Research*, 54(1):11 – 25, 2006.
- [24] D. Gupta and B. Denton. Appointment scheduling in health care: Challenges and opportunities. *IIE Transactions*, 40(9):800 – 819, 2008.
- [25] E.W. Hans, M. Houdenhoven van, and P.J.H. Hulshof. A framework for health care planning and control. Memorandum No. 19571, University of Twente, Department of Mathematical Sciences, 2011.
- [26] R. Hassin and S. Mendel. Scheduling Arrivals to Queues: A Single-Server Model with No-Shows. *Management Science*, 54(3):565 – 572, 2008.
- [27] A. Hertz and M. Widmer. Guidelines for the use of meta-heuristics in combinatorial optimization. *European Journal of Operational Research*, 151:247 – 252, 2003.
- [28] C.-J. Ho and H.-S. Lau. Minimizing Total Cost in Scheduling Outpatient Appointments. *Management Science*, 38(12):1750 – 1764, 1992.
- [29] G.C. Kaandorp and G. Koole. Optimal outpatient appointment scheduling. *Health Care Management Science*, 10:217 – 229, 2007.
- [30] KiesBeter.nl. Een CT-scan laten maken. Downloaded on 28 June 2012, from <http://www.kiesbeter.nl/ziekte-en-gezondheid/artikelen/onderzoek/ct-scan/>, n.d.
- [31] S. Kim and R.E. Giachetti. A Stochastic Mathematical Appointment Overbooking Model for Healthcare Providers to Improve Profits. *IEEE Transactions on System, Man and Cybernetics*, 36(6):1211 – 1219, 2006.

- [32] S.E. Kimes, R.B. Chase, S. Choi, P.Y. Lee, and E.N. Ngonzi. Restaurant Revenue Management: Applying Yield Management to the Restaurant Industry. *Cornell Hotel and Restaurant Administration Quarterly*, 39:32 – 39, 1998.
- [33] K. Klassen and T. Rohleder. Outpatient appointment scheduling with urgent clients in a dynamic, multi-period environment. *International Journal of Service Industry Management*, 15(2):167 – 186, 2004.
- [34] K.J. Klassen and T.R. Rohleder. Scheduling outpatient appointments in a dynamic environment. *Journal of Operations Management*, 14:83 – 101, 1996.
- [35] R. Kolisch and S. Sickinger. Providing radiology health care services to stochastic demand of different customer classes. *OR Spectrum*, 30:375 – 395, 2008.
- [36] R. Kopach, P.-C. DeLaurentis, M. Lawley, K. Muthuraman, L. Ozsen, R. Rardin, H. Wan, P. Intrevido, X. Qu, and D. Willis. Effects of clinical characteristics on successful open access scheduling. *Health Care Management Science*, 10:111 – 124, 2007.
- [37] N. Kortbeek, M.E. Zonderland, R.J. Boucherie, N. Litvak, and E.W. Hans. Designing Cyclic Appointment Schedules for Outpatient Clinics with Scheduled and Unscheduled Patient Arrivals. Memorandum No. 1968, University of Twente, Department of Applied Mathematics, 2011.
- [38] J. Kranenburg. The prospect of walk-in for the CT department of the AMC. Master’s thesis, University of Twente, 2009.
- [39] J. Lin, K. Muthuraman, and M. Lawley. Optimal and approximate algorithms for sequential clinical scheduling with no-shows. *IIE Transactions on Healthcare Systems Engineering*, 1:20 – 36, 2011.
- [40] M.L. Littman, T.L. Dean, and L.P. Kaelbling. On the Complexity of solving Markov Decision Problems. In *Proceedings of the 11th annual conference on uncertainty in artificial intelligence*, pages 394 – 402, Québec, Canada, 1995.
- [41] M. Murray and D.M. Berwick. Advanced Access: Reducing Waiting and Delays in Primary Care. *Journal of the American Medical Association*, 289(8):1035 – 1040, 2003.
- [42] C.D. O’Hare and J. Corlett. The outcomes of open-access scheduling. *Family Practice Management*, 11(2):35 – 38, 2004.
- [43] J. Patrick. A Markov decision model for determining optimal outpatient scheduling. *Health Care Management Science*, Online Publication:1–12, 2011.
- [44] J. Patrick, M.L. Puterman, and M. Queyranne. Dynamic Multipriority Patient Scheduling for a Diagnostic Resource. *Operations Research*, 56(6):1507 – 1525, 2008.
- [45] S. Pierdon, T. Charles, K. McKinley, and L. Myers. Implementing advanced access in a group practice network. *Family Practice Management*, 11(5):35 – 38, 2004.
- [46] S.D. Pizer and S.C. Prentice. What Are the Consequences of Waiting for Health Care in the Veteran Population? *Journal of General Internal Medicine*, 26(2):676 – 682, 2011.
- [47] X. Qu, R.L. Rardin, J.A.S. Williams, and D.R. Willis. Matching daily healthcare provider capacity to demand in advanced access scheduling systems. *European Journal of Operational Research*, 183:812 – 826, 2007.

- [48] A. Ratcliffe, W. Gilland, and A. Maruchek. Revenue management for outpatient appointments: joint capacity control and overbooking with class-dependent no-shows. *Flexible Services and Manufacturing Journal*, Online:1 – 33, 2011. 10.1007/s10696-011-9129-9.
- [49] L.W. Robinson and R.R. Chen. Scheduling doctors' appointments: optimal and empirically-based heuristic policies. *IIE Transactions*, 35(3):295 – 307, 2003.
- [50] T. Rohleder, P. Lewkonja, D.P. Bischak, P. Duffy, and R. Hendijani. Using simulation modeling to improve patient flow at an outpatient orthopedic clinic. *Health Care Management Science*, 14:135 – 145, 2011.
- [51] C. Sanmartin, S.E. Shortt, M.L. Barer, S. Sheps, S. Lewis, and P.W. McDonald. Waiting for medical services in Canada: lots of heat, but little light. *Canadian Medical Association Journal*, 162(9):1305 – 1310, 2000.
- [52] M. Scholtens. Visiting the CT-scan; appointment system or walk? Patient preferences and possible arrival pattern. Master's thesis, University of Twente, 2009.
- [53] S. Sickinger and R. Kolisch. The performance of a generalized Bailey-Welch rule for outpatient appointment scheduling under inpatient and emergency demand. *Health Care Management Science*, 12(4):408 – 419, 2009.
- [54] E.A. Silver, D.F. Pyke, and R. Peterson. *Inventory Management and Production Planning and Scheduling*. John Wiley and Sons, 1998.
- [55] A. Soriano. Comparison of Two Scheduling Systems. *Operations Research*, 14(3):388 – 397, 1966.
- [56] S. Su and C.-L. Shih. Managing a mixed-registration-type appointment system in outpatient clinics. *International Journal of Medical Informatics*, 70:31 – 40, 2003.
- [57] K. Talluri and G. van Ryzin. An Analysis of Bid-Price Controls for Network Revenue Management. *Management Science*, 44(11):1577 – 1593, 1998.
- [58] P.M. Vanden Bosch and D.C. Dietz. Minimizing expected waiting in a medical appointment system. *IIE Transactions*, 32:841 – 848, 2000.
- [59] P.M. Vanden Bosch, D.C. Dietz, and J.R. Simeoni. Scheduling Customer Arrivals to a Stochastic Service System. *Naval Research Logistics*, 46:549 – 559, 1999.
- [60] J.D. Welch and N.T.J. Bailey. Appointment Systems in outpatient departments. *The Lancet*, 259:1105 – 1108, 1952.

# Appendix A

## Day schedule evaluation example

In this appendix we discuss the approach to determine the performance of day schedules, as described in Kortbeek et al. [37].

Not all reserved appointment slots are always used: this is dependent on the number of patients waiting to get an appointment, which follows a certain probability distribution. Kortbeek et al. [37] introduce the following additional notation:  $\tilde{C}^d = \tilde{c}_1^d, \dots, \tilde{c}_T^d$  where  $\tilde{C}^d$  is the day schedule of day  $d$  with the realised appointments and  $\tilde{c}_t^d$  indicates how many appointments were realised on day  $d$  in time slot  $t$ . Kortbeek et al. [37] assume that patients are served First Come First Served (FCFS). The realised schedule  $\tilde{C}^d$  is always a truncated version of  $C^d$  (i.e. not all scheduled appointments should be realised in practice), the appointment schedule on day  $d$ .

We present a numerical example to clarify the evaluation of a day schedule  $C^d$ . Assume that  $k^d = 4$  and that we have to allocate these appointments to 8 time slots. The day schedule we want to evaluate is  $C^d = (1, 0, 1, 0, 0, 1, 1, 0)$ . To do this, we evaluate all possible realisation schedules. All realisation schedules have a certain probability of occurrence, based on the expected number of appointments that are present. Let  $B^d$  be the probability distribution of the appointments realised on day  $d$ , let  $P(B^d = i)$  be the probability that the realised number of appointments on day  $d$  equals  $i$  and let  $\nu^d$  represent the number of deferred patients on day  $d$ . Table A.1 displays all possible realised day schedules and their performance. Here we added the subscript  $i$  to  $\tilde{c}^d$  and  $\nu^d$  to indicate the realised day schedule and number of deferred patients when the realised number of scheduled patients is equal to  $i$ . The performance of day schedule  $C^d$  is the expected number of deferred patients. In the numerical example in table A.1 the expected number of deferred patients is 1.07, which is calculated by:  $\sum_{i=0}^{\infty} P(B^d = i) * \nu_i^d$ . In our numerical example we assume that no more than 4 patients can arrive, to keep the example understandable.

$i$	$P(B^d = i)$	$\tilde{c}_i^d$	$\nu_i^d$
0	0.05	(0, 0, 0, 0, 0, 0, 0, 0)	0.00
1	0.20	(1, 0, 0, 0, 0, 0, 0, 0)	0.20
2	0.10	(1, 0, 1, 0, 0, 0, 0, 0)	0.80
3	0.15	(1, 0, 1, 0, 0, 1, 0, 0)	1.00
4	0.50	(1, 0, 1, 0, 0, 1, 1, 0)	1.60

Table A.1: Numerical example of day schedule evaluation

## Appendix B

# Arrival rates test setting

In this appendix we present the arrival rate patterns as used in our tests. Table B.1 presents the walk-in arrival rates corresponding to pattern 1, Table B.2 presents the walk-in arrival rates corresponding to pattern 2 and Table B.3 presents the walk-in arrival rates corresponding to pattern 3. Table B.4 presents the initial appointment request rates of the three patterns.

$\chi_t^d$	$t$								
$d$	1	2	3	4	5	6	7	8	Total
1	0.30	0.60	1.00	1.40	1.40	1.00	0.55	0.25	6.50
2	1.10	1.00	0.90	0.80	0.70	0.60	0.50	0.40	6.00
3	0.15	0.30	0.45	0.60	0.60	0.45	0.30	0.15	3.00
4	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.80
5	0.30	0.90	1.50	1.00	0.30	0.75	0.65	0.30	5.70

Table B.1: Walk-in arrival rates pattern 1

$\chi_t^d$	$t$								
$d$	1	2	3	4	5	6	7	8	Total
1	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	3.60
2	0.90	1.20	1.30	1.20	1.00	0.80	0.60	0.40	7.40
3	0.70	1.00	0.70	1.40	0.70	0.90	1.00	0.90	7.30
4	0.00	0.15	0.20	0.20	0.20	0.20	0.15	0.00	1.10
5	0.20	0.40	0.40	0.20	0.40	0.50	0.40	0.10	2.60

Table B.2: Walk-in arrival rates pattern 2

$\chi_t^d$	$t$								
$d$	1	2	3	4	5	6	7	8	Total
1	0.20	0.50	0.80	0.50	0.50	0.80	0.50	0.20	4.00
2	0.10	0.30	0.60	1.00	1.00	0.60	0.30	0.10	4.00
3	0.20	1.00	1.20	0.20	0.20	0.80	1.20	0.20	5.00
4	0.10	0.10	0.10	3.00	0.10	0.10	0.10	0.10	3.70
5	0.30	0.70	0.30	0.70	0.30	0.70	0.30	0.70	4.00

Table B.3: Walk-in arrival rates pattern 3



$d$	$\lambda^d$		
	Pattern 1	Pattern 2	Pattern 3
1	5	2	3
2	0	3	3
3	2	4	3
4	0	3	3
5	7	2	3

Table B.4: Appointment request rates

## Appendix C

# Results of capacity cycle generation tests

In this appendix we present detailed results of the tests we performed on our capacity cycle heuristics in combination with local search for capacity cycles, while using complete enumeration for day schedules.

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
1	1	5	8	2	0.15	10, 95%	P1	<b>CE+CE</b>	5.969	-	32220.0	537.00
2	1	5	8	4	0.15	10, 95%	P1	<b>CE+CE</b>	4.362	-	31620.0	527.00
3	1	5	8	2	0.00	10, 95%	P1	<b>CE+CE</b>	6.890	-	45600.0	760.00
4	1	5	8	2	0.15	15, 95%	P1	<b>CE+CE</b>	5.639	-	31800.0	530.00
5	1	5	8	2	0.15	5, 95%	P1	<b>CE+CE</b>	6.540	-	54660.0	911.00
6	1	5	8	4	0.15	15, 95%	P1	<b>CE+CE</b>	3.966	-	31467.5	524.46
7	1	5	8	2	0.00	5, 95%	P1	<b>CE+CE</b>	7.570	-	56284.7	938.08
8	1	5	8	2	0.00	15, 95%	P1	<b>CE+CE</b>	6.477	-	43319.9	722.00
9	1	5	8	4	0.00	5, 95%	P1	<b>CE+CE</b>	5.558	-	58078.7	967.98
10	1	5	8	4	0.15	5, 95%	P1	<b>CE+CE</b>	4.548	-	36738.8	612.31
11	1	5	8	4	0.00	10, 95%	P1	<b>CE+CE</b>	4.915	-	48159.9	802.66
12	1	5	8	4	0.00	15, 95%	P1	<b>CE+CE</b>	4.666	-	31527.7	525.46
13	1	5	8	2	0.15	10, 95%	P2	<b>CE+CE</b>	6.328	-	28440.0	474.00
14	1	5	8	4	0.15	10, 95%	P2	<b>CE+CE</b>	4.668	-	23040.0	384.00
15	1	5	8	2	0.00	10, 95%	P2	<b>CE+CE</b>	6.797	-	28020.0	467.00
16	1	5	8	2	0.15	15, 95%	P2	<b>CE+CE</b>	6.328	-	28320.0	472.00
17	1	5	8	2	0.15	5, 95%	P2	<b>CE+CE</b>	6.767	-	30210.0	503.50
18	1	5	8	4	0.15	15, 95%	P2	<b>CE+CE</b>	4.669	-	21527.3	358.79
19	1	5	8	2	0.00	5, 95%	P2	<b>CE+CE</b>	7.692	-	28047.0	467.45
20	1	5	8	2	0.00	15, 95%	P2	<b>CE+CE</b>	6.797	-	27404.6	456.74
21	1	5	8	4	0.00	5, 95%	P2	<b>CE+CE</b>	6.253	-	46894.9	781.58
22	1	5	8	4	0.15	5, 95%	P2	<b>CE+CE</b>	5.292	-	26061.3	434.35
23	1	5	8	4	0.00	10, 95%	P2	<b>CE+CE</b>	5.373	-	22354.1	372.57
24	1	5	8	4	0.00	15, 95%	P2	<b>CE+CE</b>	5.373	-	22138.7	368.98
25	1	5	8	2	0.15	10, 95%	P3	<b>CE+CE</b>	5.511	-	15869.3	264.49
26	1	5	8	4	0.15	10, 95%	P3	<b>CE+CE</b>	3.421	-	13176.7	219.61
27	1	5	8	2	0.00	10, 95%	P3	<b>CE+CE</b>	6.446	-	17277.1	287.95
28	1	5	8	2	0.15	15, 95%	P3	<b>CE+CE</b>	5.511	-	17481.0	291.35
29	1	5	8	2	0.15	5, 95%	P3	<b>CE+CE</b>	6.308	-	16379.5	272.99
30	1	5	8	4	0.15	15, 95%	P3	<b>CE+CE</b>	3.421	-	13986.1	233.10
31	1	5	8	2	0.00	5, 95%	P3	<b>CE+CE</b>	7.450	-	20237.2	337.29
32	1	5	8	2	0.00	15, 95%	P3	<b>CE+CE</b>	6.446	-	19297.4	321.62
33	1	5	8	4	0.00	5, 95%	P3	<b>CE+CE</b>	4.928	-	18237.4	303.96
34	1	5	8	4	0.15	5, 95%	P3	<b>CE+CE</b>	3.842	-	17260.1	287.67
35	1	5	8	4	0.00	10, 95%	P3	<b>CE+CE</b>	4.402	-	17574.5	292.91

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
36	1	5	8	4	0.00	15, 95%	P3	<b>CE+CE</b>	4.402	-	19426.0	323.77
37	1	5	8	2	0.15	10, 95%	P1	<b>H1+CE</b>	6.007	0.65%	2.9	0.05
38	1	5	8	4	0.15	10, 95%	P1	<b>H1+CE</b>	4.362	0.00%	3.6	0.06
39	1	5	8	2	0.00	10, 95%	P1	<b>H1+CE</b>	6.913	0.34%	3.7	0.06
40	1	5	8	2	0.15	15, 95%	P1	<b>H1+CE</b>	5.639	0.00%	2.2	0.04
41	1	5	8	2	0.15	5, 95%	P1	<b>H1+CE</b>	6.801	3.99%	4.8	0.08
42	1	5	8	4	0.15	15, 95%	P1	<b>H1+CE</b>	3.966	0.00%	2.2	0.04
43	1	5	8	2	0.00	5, 95%	P1	<b>H1+CE</b>	7.739	2.23%	5.8	0.10
44	1	5	8	2	0.00	15, 95%	P1	<b>H1+CE</b>	6.536	0.91%	2.4	0.04
45	1	5	8	4	0.00	5, 95%	P1	<b>H1+CE</b>	5.752	3.49%	5.1	0.08
46	1	5	8	4	0.15	5, 95%	P1	<b>H1+CE</b>	4.663	2.53%	3.9	0.06
47	1	5	8	4	0.00	10, 95%	P1	<b>H1+CE</b>	4.915	0.00%	3.6	0.06
48	1	5	8	4	0.00	15, 95%	P1	<b>H1+CE</b>	4.761	2.04%	2.4	0.04
49	1	5	8	2	0.15	10, 95%	P2	<b>H1+CE</b>	6.328	0.00%	2.0	0.03
50	1	5	8	4	0.15	10, 95%	P2	<b>H1+CE</b>	4.669	0.02%	2.0	0.03
51	1	5	8	2	0.00	10, 95%	P2	<b>H1+CE</b>	6.798	0.00%	1.9	0.03
52	1	5	8	2	0.15	15, 95%	P2	<b>H1+CE</b>	6.328	0.00%	1.9	0.03
53	1	5	8	2	0.15	5, 95%	P2	<b>H1+CE</b>	7.105	4.99%	4.9	0.08
54	1	5	8	4	0.15	15, 95%	P2	<b>H1+CE</b>	4.669	0.00%	2.2	0.04
55	1	5	8	2	0.00	5, 95%	P2	<b>H1+CE</b>	7.788	1.25%	3.7	0.06
56	1	5	8	2	0.00	15, 95%	P2	<b>H1+CE</b>	6.798	0.00%	1.9	0.03
57	1	5	8	4	0.00	5, 95%	P2	<b>H1+CE</b>	6.404	2.42%	3.9	0.06
58	1	5	8	4	0.15	5, 95%	P2	<b>H1+CE</b>	5.550	4.87%	3.6	0.06
59	1	5	8	4	0.00	10, 95%	P2	<b>H1+CE</b>	5.455	1.54%	2.1	0.04
60	1	5	8	4	0.00	15, 95%	P2	<b>H1+CE</b>	5.455	1.54%	2.1	0.04
61	1	5	8	2	0.15	10, 95%	P3	<b>H1+CE</b>	5.607	1.74%	2.2	0.04
62	1	5	8	4	0.15	10, 95%	P3	<b>H1+CE</b>	3.421	0.00%	2.4	0.04
63	1	5	8	2	0.00	10, 95%	P3	<b>H1+CE</b>	6.586	2.16%	2.3	0.04
64	1	5	8	2	0.15	15, 95%	P3	<b>H1+CE</b>	5.607	1.74%	2.1	0.03
65	1	5	8	2	0.15	5, 95%	P3	<b>H1+CE</b>	6.308	0.00%	3.5	0.06
66	1	5	8	4	0.15	15, 95%	P3	<b>H1+CE</b>	3.421	0.00%	2.4	0.04
67	1	5	8	2	0.00	5, 95%	P3	<b>H1+CE</b>	7.542	1.23%	4.2	0.07
68	1	5	8	2	0.00	15, 95%	P3	<b>H1+CE</b>	6.586	2.16%	2.3	0.04
69	1	5	8	4	0.00	5, 95%	P3	<b>H1+CE</b>	5.200	5.52%	4.6	0.08
70	1	5	8	4	0.15	5, 95%	P3	<b>H1+CE</b>	3.928	2.25%	3.3	0.05
71	1	5	8	4	0.00	10, 95%	P3	<b>H1+CE</b>	4.402	0.00%	2.7	0.05
72	1	5	8	4	0.00	15, 95%	P3	<b>H1+CE</b>	4.402	0.00%	2.7	0.04
73	1	5	8	2	0.15	10, 95%	P1	<b>H2+CE</b>	6.007	0.65%	3.0	0.05
74	1	5	8	4	0.15	10, 95%	P1	<b>H2+CE</b>	4.362	0.00%	3.7	0.06
75	1	5	8	2	0.00	10, 95%	P1	<b>H2+CE</b>	6.913	0.34%	4.2	0.07
76	1	5	8	2	0.15	15, 95%	P1	<b>H2+CE</b>	5.639	0.00%	2.2	0.04
77	1	5	8	2	0.15	5, 95%	P1	<b>H2+CE</b>	6.801	3.99%	5.0	0.08
78	1	5	8	4	0.15	15, 95%	P1	<b>H2+CE</b>	3.966	0.00%	2.2	0.04
79	1	5	8	2	0.00	5, 95%	P1	<b>H2+CE</b>	7.739	2.23%	6.5	0.11
80	1	5	8	2	0.00	15, 95%	P1	<b>H2+CE</b>	6.536	0.91%	2.5	0.04
81	1	5	8	4	0.00	5, 95%	P1	<b>H2+CE</b>	5.752	3.49%	5.1	0.08
82	1	5	8	4	0.15	5, 95%	P1	<b>H2+CE</b>	4.663	2.53%	3.9	0.07
83	1	5	8	4	0.00	10, 95%	P1	<b>H2+CE</b>	4.915	0.00%	3.7	0.06
84	1	5	8	4	0.00	15, 95%	P1	<b>H2+CE</b>	4.761	2.04%	2.4	0.04
85	1	5	8	2	0.15	10, 95%	P2	<b>H2+CE</b>	6.338	0.16%	2.0	0.03
86	1	5	8	4	0.15	10, 95%	P2	<b>H2+CE</b>	4.793	2.68%	2.0	0.03
87	1	5	8	2	0.00	10, 95%	P2	<b>H2+CE</b>	6.863	0.96%	2.2	0.04
88	1	5	8	2	0.15	15, 95%	P2	<b>H2+CE</b>	6.338	0.16%	1.9	0.03
89	1	5	8	2	0.15	5, 95%	P2	<b>H2+CE</b>	7.105	4.99%	5.1	0.08
90	1	5	8	4	0.15	15, 95%	P2	<b>H2+CE</b>	4.793	2.65%	2.0	0.03
91	1	5	8	2	0.00	5, 95%	P2	<b>H2+CE</b>	7.788	1.25%	3.8	0.06
92	1	5	8	2	0.00	15, 95%	P2	<b>H2+CE</b>	6.863	0.96%	2.0	0.03
93	1	5	8	4	0.00	5, 95%	P2	<b>H2+CE</b>	6.404	2.42%	4.6	0.08
94	1	5	8	4	0.15	5, 95%	P2	<b>H2+CE</b>	5.368	1.43%	3.4	0.06
95	1	5	8	4	0.00	10, 95%	P2	<b>H2+CE</b>	5.455	1.54%	2.2	0.04

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
96	1	5	8	4	0.00	15, 95%	P2	<b>H2+CE</b>	5.455	1.54%	2.1	0.04
97	1	5	8	2	0.15	10, 95%	P3	<b>H2+CE</b>	5.511	0.00%	2.2	0.04
98	1	5	8	4	0.15	10, 95%	P3	<b>H2+CE</b>	3.421	0.00%	2.5	0.04
99	1	5	8	2	0.00	10, 95%	P3	<b>H2+CE</b>	6.502	0.87%	2.3	0.04
100	1	5	8	2	0.15	15, 95%	P3	<b>H2+CE</b>	5.511	0.00%	2.1	0.03
101	1	5	8	2	0.15	5, 95%	P3	<b>H2+CE</b>	6.308	0.00%	3.5	0.06
102	1	5	8	4	0.15	15, 95%	P3	<b>H2+CE</b>	3.421	0.00%	2.5	0.04
103	1	5	8	2	0.00	5, 95%	P3	<b>H2+CE</b>	7.542	1.23%	4.1	0.07
104	1	5	8	2	0.00	15, 95%	P3	<b>H2+CE</b>	6.502	0.87%	2.4	0.04
105	1	5	8	4	0.00	5, 95%	P3	<b>H2+CE</b>	4.972	0.91%	4.1	0.07
106	1	5	8	4	0.15	5, 95%	P3	<b>H2+CE</b>	3.872	0.80%	3.5	0.06
107	1	5	8	4	0.00	10, 95%	P3	<b>H2+CE</b>	4.402	0.00%	2.6	0.04
108	1	5	8	4	0.00	15, 95%	P3	<b>H2+CE</b>	4.402	0.00%	2.6	0.04
109	1	5	8	2	0.15	10, 95%	P1	<b>H1+LS-CC+CE</b>	5.969	0.00%	37.4	0.62
110	1	5	8	4	0.15	10, 95%	P1	<b>H1+LS-CC+CE</b>	4.362	0.00%	34.9	0.58
111	1	5	8	2	0.00	10, 95%	P1	<b>H1+LS-CC+CE</b>	6.890	0.00%	56.3	0.94
112	1	5	8	2	0.15	15, 95%	P1	<b>H1+LS-CC+CE</b>	5.639	0.00%	30.2	0.50
113	1	5	8	2	0.15	5, 95%	P1	<b>H1+LS-CC+CE</b>	6.591	0.79%	54.8	0.91
114	1	5	8	4	0.15	15, 95%	P1	<b>H1+LS-CC+CE</b>	3.966	0.00%	21.9	0.37
115	1	5	8	2	0.00	5, 95%	P1	<b>H1+LS-CC+CE</b>	7.570	0.00%	74.1	1.24
116	1	5	8	2	0.00	15, 95%	P1	<b>H1+LS-CC+CE</b>	6.477	0.00%	55.0	0.92
117	1	5	8	4	0.00	5, 95%	P1	<b>H1+LS-CC+CE</b>	5.567	0.16%	57.2	0.95
118	1	5	8	4	0.15	5, 95%	P1	<b>H1+LS-CC+CE</b>	4.548	0.00%	54.4	0.91
119	1	5	8	4	0.00	10, 95%	P1	<b>H1+LS-CC+CE</b>	4.915	0.00%	35.7	0.59
120	1	5	8	4	0.00	15, 95%	P1	<b>H1+LS-CC+CE</b>	4.761	2.04%	37.0	0.62
121	1	5	8	2	0.15	10, 95%	P2	<b>H1+LS-CC+CE</b>	6.328	0.00%	21.9	0.37
122	1	5	8	4	0.15	10, 95%	P2	<b>H1+LS-CC+CE</b>	4.669	0.02%	17.6	0.29
123	1	5	8	2	0.00	10, 95%	P2	<b>H1+LS-CC+CE</b>	6.797	0.00%	21.9	0.37
124	1	5	8	2	0.15	15, 95%	P2	<b>H1+LS-CC+CE</b>	6.328	0.00%	21.5	0.36
125	1	5	8	2	0.15	5, 95%	P2	<b>H1+LS-CC+CE</b>	7.105	4.99%	39.1	0.65
126	1	5	8	4	0.15	15, 95%	P2	<b>H1+LS-CC+CE</b>	4.669	0.00%	17.3	0.29
127	1	5	8	2	0.00	5, 95%	P2	<b>H1+LS-CC+CE</b>	7.788	1.25%	34.9	0.58
128	1	5	8	2	0.00	15, 95%	P2	<b>H1+LS-CC+CE</b>	6.797	0.00%	21.6	0.36
129	1	5	8	4	0.00	5, 95%	P2	<b>H1+LS-CC+CE</b>	6.253	0.00%	48.4	0.81
130	1	5	8	4	0.15	5, 95%	P2	<b>H1+LS-CC+CE</b>	5.292	0.00%	28.1	0.47
131	1	5	8	4	0.00	10, 95%	P2	<b>H1+LS-CC+CE</b>	5.373	0.00%	25.1	0.42
132	1	5	8	4	0.00	15, 95%	P2	<b>H1+LS-CC+CE</b>	5.373	0.00%	20.3	0.34
133	1	5	8	2	0.15	10, 95%	P3	<b>H1+LS-CC+CE</b>	5.511	0.00%	23.0	0.38
134	1	5	8	4	0.15	10, 95%	P3	<b>H1+LS-CC+CE</b>	3.421	0.00%	18.2	0.30
135	1	5	8	2	0.00	10, 95%	P3	<b>H1+LS-CC+CE</b>	6.483	0.58%	28.2	0.47
136	1	5	8	2	0.15	15, 95%	P3	<b>H1+LS-CC+CE</b>	5.511	0.00%	25.5	0.42
137	1	5	8	2	0.15	5, 95%	P3	<b>H1+LS-CC+CE</b>	6.308	0.00%	35.0	0.58
138	1	5	8	4	0.15	15, 95%	P3	<b>H1+LS-CC+CE</b>	3.421	0.00%	16.4	0.27
139	1	5	8	2	0.00	5, 95%	P3	<b>H1+LS-CC+CE</b>	7.450	0.00%	46.3	0.77
140	1	5	8	2	0.00	15, 95%	P3	<b>H1+LS-CC+CE</b>	6.483	0.58%	31.3	0.52
141	1	5	8	4	0.00	5, 95%	P3	<b>H1+LS-CC+CE</b>	4.955	0.55%	28.1	0.47
142	1	5	8	4	0.15	5, 95%	P3	<b>H1+LS-CC+CE</b>	3.872	0.80%	27.6	0.46
143	1	5	8	4	0.00	10, 95%	P3	<b>H1+LS-CC+CE</b>	4.402	0.00%	23.4	0.39
144	1	5	8	4	0.00	15, 95%	P3	<b>H1+LS-CC+CE</b>	4.402	0.00%	25.9	0.43
145	1	5	8	2	0.15	10, 95%	P1	<b>H2+LS-CC+CE</b>	5.969	0.00%	37.5	0.63
146	1	5	8	4	0.15	10, 95%	P1	<b>H2+LS-CC+CE</b>	4.362	0.00%	33.8	0.56
147	1	5	8	2	0.00	10, 95%	P1	<b>H2+LS-CC+CE</b>	6.890	0.00%	56.6	0.94
148	1	5	8	2	0.15	15, 95%	P1	<b>H2+LS-CC+CE</b>	5.639	0.00%	29.6	0.49
149	1	5	8	2	0.15	5, 95%	P1	<b>H2+LS-CC+CE</b>	6.591	0.79%	55.3	0.92
150	1	5	8	4	0.15	15, 95%	P1	<b>H2+LS-CC+CE</b>	3.966	0.00%	24.1	0.40
151	1	5	8	2	0.00	5, 95%	P1	<b>H2+LS-CC+CE</b>	7.570	0.00%	72.9	1.21
152	1	5	8	2	0.00	15, 95%	P1	<b>H2+LS-CC+CE</b>	6.477	0.00%	47.0	0.78
153	1	5	8	4	0.00	5, 95%	P1	<b>H2+LS-CC+CE</b>	5.567	0.16%	49.6	0.83
154	1	5	8	4	0.15	5, 95%	P1	<b>H2+LS-CC+CE</b>	4.548	0.00%	50.1	0.84
155	1	5	8	4	0.00	10, 95%	P1	<b>H2+LS-CC+CE</b>	4.915	0.00%	35.0	0.58

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
156	1	5	8	4	0.00	15, 95%	P1	<b>H2+LS-CC+CE</b>	4.761	2.04%	34.8	0.58
157	1	5	8	2	0.15	10, 95%	P2	<b>H2+LS-CC+CE</b>	6.338	0.16%	25.0	0.42
158	1	5	8	4	0.15	10, 95%	P2	<b>H2+LS-CC+CE</b>	4.669	0.02%	16.1	0.27
159	1	5	8	2	0.00	10, 95%	P2	<b>H2+LS-CC+CE</b>	6.863	0.96%	25.3	0.42
160	1	5	8	2	0.15	15, 95%	P2	<b>H2+LS-CC+CE</b>	6.338	0.16%	24.4	0.41
161	1	5	8	2	0.15	5, 95%	P2	<b>H2+LS-CC+CE</b>	7.105	4.99%	38.6	0.64
162	1	5	8	4	0.15	15, 95%	P2	<b>H2+LS-CC+CE</b>	4.669	0.00%	16.3	0.27
163	1	5	8	2	0.00	5, 95%	P2	<b>H2+LS-CC+CE</b>	7.788	1.25%	34.9	0.58
164	1	5	8	2	0.00	15, 95%	P2	<b>H2+LS-CC+CE</b>	6.863	0.96%	24.5	0.41
165	1	5	8	4	0.00	5, 95%	P2	<b>H2+LS-CC+CE</b>	6.253	0.00%	48.1	0.80
166	1	5	8	4	0.15	5, 95%	P2	<b>H2+LS-CC+CE</b>	5.366	1.39%	30.5	0.51
167	1	5	8	4	0.00	10, 95%	P2	<b>H2+LS-CC+CE</b>	5.373	0.00%	27.7	0.46
168	1	5	8	4	0.00	15, 95%	P2	<b>H2+LS-CC+CE</b>	5.373	0.00%	19.9	0.33
169	1	5	8	2	0.15	10, 95%	P3	<b>H2+LS-CC+CE</b>	5.511	0.00%	23.1	0.39
170	1	5	8	4	0.15	10, 95%	P3	<b>H2+LS-CC+CE</b>	3.421	0.00%	16.2	0.27
171	1	5	8	2	0.00	10, 95%	P3	<b>H2+LS-CC+CE</b>	6.446	0.00%	28.7	0.48
172	1	5	8	2	0.15	15, 95%	P3	<b>H2+LS-CC+CE</b>	5.511	0.00%	28.0	0.47
173	1	5	8	2	0.15	5, 95%	P3	<b>H2+LS-CC+CE</b>	6.308	0.00%	35.6	0.59
174	1	5	8	4	0.15	15, 95%	P3	<b>H2+LS-CC+CE</b>	3.421	0.00%	16.5	0.27
175	1	5	8	2	0.00	5, 95%	P3	<b>H2+LS-CC+CE</b>	7.450	0.00%	46.6	0.78
176	1	5	8	2	0.00	15, 95%	P3	<b>H2+LS-CC+CE</b>	6.446	0.00%	32.8	0.55
177	1	5	8	4	0.00	5, 95%	P3	<b>H2+LS-CC+CE</b>	4.928	0.00%	29.5	0.49
178	1	5	8	4	0.15	5, 95%	P3	<b>H2+LS-CC+CE</b>	3.842	0.00%	24.7	0.41
179	1	5	8	4	0.00	10, 95%	P3	<b>H2+LS-CC+CE</b>	4.402	0.00%	23.1	0.38
180	1	5	8	4	0.00	15, 95%	P3	<b>H2+LS-CC+CE</b>	4.402	0.00%	26.9	0.45

## Appendix D

# Results of day schedule generation tests

In this appendix we present detailed results of the tests we performed on our capacity cycle heuristics, in combination with our day schedule heuristics and the local search techniques for capacity cycles and day schedules.

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
1	1	5	8	2	0.15	10, 95%	P1	<b>CE+CE</b>	5.969	-	32220.0	537.00
2	1	5	8	4	0.15	10, 95%	P1	<b>CE+CE</b>	4.362	-	31620.0	527.00
3	1	5	8	2	0.00	10, 95%	P1	<b>CE+CE</b>	6.890	-	45600.0	760.00
4	1	5	8	2	0.15	15, 95%	P1	<b>CE+CE</b>	5.639	-	31800.0	530.00
5	1	5	8	2	0.15	5, 95%	P1	<b>CE+CE</b>	6.540	-	54660.0	911.00
6	1	5	8	4	0.15	15, 95%	P1	<b>CE+CE</b>	3.966	-	31467.5	524.46
7	1	5	8	2	0.00	5, 95%	P1	<b>CE+CE</b>	7.570	-	56284.7	938.08
8	1	5	8	2	0.00	15, 95%	P1	<b>CE+CE</b>	6.477	-	43319.9	722.00
9	1	5	8	4	0.00	5, 95%	P1	<b>CE+CE</b>	5.558	-	58078.7	967.98
10	1	5	8	4	0.15	5, 95%	P1	<b>CE+CE</b>	4.548	-	36738.8	612.31
11	1	5	8	4	0.00	10, 95%	P1	<b>CE+CE</b>	4.915	-	48159.9	802.66
12	1	5	8	4	0.00	15, 95%	P1	<b>CE+CE</b>	4.666	-	31527.7	525.46
13	1	5	8	2	0.15	10, 95%	P2	<b>CE+CE</b>	6.328	-	28440.0	474.00
14	1	5	8	4	0.15	10, 95%	P2	<b>CE+CE</b>	4.668	-	23040.0	384.00
15	1	5	8	2	0.00	10, 95%	P2	<b>CE+CE</b>	6.797	-	28020.0	467.00
16	1	5	8	2	0.15	15, 95%	P2	<b>CE+CE</b>	6.328	-	28320.0	472.00
17	1	5	8	2	0.15	5, 95%	P2	<b>CE+CE</b>	6.767	-	30210.0	503.50
18	1	5	8	4	0.15	15, 95%	P2	<b>CE+CE</b>	4.669	-	21527.3	358.79
19	1	5	8	2	0.00	5, 95%	P2	<b>CE+CE</b>	7.692	-	28047.0	467.45
20	1	5	8	2	0.00	15, 95%	P2	<b>CE+CE</b>	6.797	-	27404.6	456.74
21	1	5	8	4	0.00	5, 95%	P2	<b>CE+CE</b>	6.253	-	46894.9	781.58
22	1	5	8	4	0.15	5, 95%	P2	<b>CE+CE</b>	5.292	-	26061.3	434.35
23	1	5	8	4	0.00	10, 95%	P2	<b>CE+CE</b>	5.373	-	22354.1	372.57
24	1	5	8	4	0.00	15, 95%	P2	<b>CE+CE</b>	5.373	-	22138.7	368.98
25	1	5	8	2	0.15	10, 95%	P3	<b>CE+CE</b>	5.511	-	15869.3	264.49
26	1	5	8	4	0.15	10, 95%	P3	<b>CE+CE</b>	3.421	-	13176.7	219.61
27	1	5	8	2	0.00	10, 95%	P3	<b>CE+CE</b>	6.446	-	17277.1	287.95
28	1	5	8	2	0.15	15, 95%	P3	<b>CE+CE</b>	5.511	-	17481.0	291.35
29	1	5	8	2	0.15	5, 95%	P3	<b>CE+CE</b>	6.308	-	16379.5	272.99
30	1	5	8	4	0.15	15, 95%	P3	<b>CE+CE</b>	3.421	-	13986.1	233.10
31	1	5	8	2	0.00	5, 95%	P3	<b>CE+CE</b>	7.450	-	20237.2	337.29
32	1	5	8	2	0.00	15, 95%	P3	<b>CE+CE</b>	6.446	-	19297.4	321.62
33	1	5	8	4	0.00	5, 95%	P3	<b>CE+CE</b>	4.928	-	18237.4	303.96

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
34	1	5	8	4	0.15	5, 95%	P3	<b>CE+CE</b>	3.842	-	17260.1	287.67
35	1	5	8	4	0.00	10, 95%	P3	<b>CE+CE</b>	4.402	-	17574.5	292.91
36	1	5	8	4	0.00	15, 95%	P3	<b>CE+CE</b>	4.402	-	19426.0	323.77
37	1	5	8	2	0.15	10, 95%	P1	<b>H1+H3+LS-RS</b>	6.007	0.65%	2.0	0.03
38	1	5	8	4	0.15	10, 95%	P1	<b>H1+H3+LS-RS</b>	4.362	0.00%	2.1	0.04
39	1	5	8	2	0.00	10, 95%	P1	<b>H1+H3+LS-RS</b>	6.913	0.34%	2.8	0.05
40	1	5	8	2	0.15	15, 95%	P1	<b>H1+H3+LS-RS</b>	5.639	0.00%	1.2	0.02
41	1	5	8	2	0.15	5, 95%	P1	<b>H1+H3+LS-RS</b>	6.801	3.99%	4.2	0.07
42	1	5	8	4	0.15	15, 95%	P1	<b>H1+H3+LS-RS</b>	4.191	5.70%	1.6	0.03
43	1	5	8	2	0.00	5, 95%	P1	<b>H1+H3+LS-RS</b>	7.739	2.23%	5.2	0.09
44	1	5	8	2	0.00	15, 95%	P1	<b>H1+H3+LS-RS</b>	6.584	1.65%	1.4	0.02
45	1	5	8	4	0.00	5, 95%	P1	<b>H1+H3+LS-RS</b>	5.752	3.49%	4.7	0.08
46	1	5	8	4	0.15	5, 95%	P1	<b>H1+H3+LS-RS</b>	4.663	2.53%	3.2	0.05
47	1	5	8	4	0.00	10, 95%	P1	<b>H1+H3+LS-RS</b>	4.915	0.00%	2.6	0.04
48	1	5	8	4	0.00	15, 95%	P1	<b>H1+H3+LS-RS</b>	4.761	2.04%	1.5	0.02
49	1	5	8	2	0.15	10, 95%	P2	<b>H1+H3+LS-RS</b>	6.328	0.00%	1.3	0.02
50	1	5	8	4	0.15	10, 95%	P2	<b>H1+H3+LS-RS</b>	4.669	0.02%	1.0	0.02
51	1	5	8	2	0.00	10, 95%	P2	<b>H1+H3+LS-RS</b>	6.798	0.00%	1.2	0.02
52	1	5	8	2	0.15	15, 95%	P2	<b>H1+H3+LS-RS</b>	6.328	0.00%	1.2	0.02
53	1	5	8	2	0.15	5, 95%	P2	<b>H1+H3+LS-RS</b>	7.105	4.99%	3.8	0.06
54	1	5	8	4	0.15	15, 95%	P2	<b>H1+H3+LS-RS</b>	4.669	0.00%	1.0	0.02
55	1	5	8	2	0.00	5, 95%	P2	<b>H1+H3+LS-RS</b>	7.788	1.25%	3.4	0.06
56	1	5	8	2	0.00	15, 95%	P2	<b>H1+H3+LS-RS</b>	6.797	0.00%	1.1	0.02
57	1	5	8	4	0.00	5, 95%	P2	<b>H1+H3+LS-RS</b>	6.404	2.42%	2.8	0.05
58	1	5	8	4	0.15	5, 95%	P2	<b>H1+H3+LS-RS</b>	5.550	4.87%	2.6	0.04
59	1	5	8	4	0.00	10, 95%	P2	<b>H1+H3+LS-RS</b>	5.455	1.54%	1.3	0.02
60	1	5	8	4	0.00	15, 95%	P2	<b>H1+H3+LS-RS</b>	5.455	1.54%	1.3	0.02
61	1	5	8	2	0.15	10, 95%	P3	<b>H1+H3+LS-RS</b>	5.607	1.74%	1.1	0.02
62	1	5	8	4	0.15	10, 95%	P3	<b>H1+H3+LS-RS</b>	3.452	0.90%	0.8	0.01
63	1	5	8	2	0.00	10, 95%	P3	<b>H1+H3+LS-RS</b>	6.586	2.16%	1.7	0.03
64	1	5	8	2	0.15	15, 95%	P3	<b>H1+H3+LS-RS</b>	5.607	1.74%	1.0	0.02
65	1	5	8	2	0.15	5, 95%	P3	<b>H1+H3+LS-RS</b>	6.312	0.06%	2.5	0.04
66	1	5	8	4	0.15	15, 95%	P3	<b>H1+H3+LS-RS</b>	3.421	0.00%	0.8	0.01
67	1	5	8	2	0.00	5, 95%	P3	<b>H1+H3+LS-RS</b>	7.542	1.23%	3.0	0.05
68	1	5	8	2	0.00	15, 95%	P3	<b>H1+H3+LS-RS</b>	6.586	2.16%	1.3	0.02
69	1	5	8	4	0.00	5, 95%	P3	<b>H1+H3+LS-RS</b>	5.200	5.52%	3.8	0.06
70	1	5	8	4	0.15	5, 95%	P3	<b>H1+H3+LS-RS</b>	3.928	2.25%	2.0	0.03
71	1	5	8	4	0.00	10, 95%	P3	<b>H1+H3+LS-RS</b>	4.402	0.00%	1.4	0.02
72	1	5	8	4	0.00	15, 95%	P3	<b>H1+H3+LS-RS</b>	4.402	0.00%	1.4	0.02
73	1	5	8	2	0.15	10, 95%	P1	<b>H1+H4+LS-RS</b>	6.007	0.65%	2.4	0.04
74	1	5	8	4	0.15	10, 95%	P1	<b>H1+H4+LS-RS</b>	4.362	0.00%	2.7	0.05
75	1	5	8	2	0.00	10, 95%	P1	<b>H1+H4+LS-RS</b>	6.914	0.34%	2.6	0.04
76	1	5	8	2	0.15	15, 95%	P1	<b>H1+H4+LS-RS</b>	6.007	6.54%	1.8	0.03
77	1	5	8	2	0.15	5, 95%	P1	<b>H1+H4+LS-RS</b>	7.010	7.18%	3.1	0.05
78	1	5	8	4	0.15	15, 95%	P1	<b>H1+H4+LS-RS</b>	4.191	5.70%	1.1	0.02
79	1	5	8	2	0.00	5, 95%	P1	<b>H1+H4+LS-RS</b>	7.739	2.23%	4.8	0.08
80	1	5	8	2	0.00	15, 95%	P1	<b>H1+H4+LS-RS</b>	6.584	1.65%	2.8	0.05
81	1	5	8	4	0.00	5, 95%	P1	<b>H1+H4+LS-RS</b>	6.166	10.94%	5.8	0.10
82	1	5	8	4	0.15	5, 95%	P1	<b>H1+H4+LS-RS</b>	4.663	2.53%	3.1	0.05
83	1	5	8	4	0.00	10, 95%	P1	<b>H1+H4+LS-RS</b>	5.311	8.07%	3.2	0.05
84	1	5	8	4	0.00	15, 95%	P1	<b>H1+H4+LS-RS</b>	4.761	2.04%	1.7	0.03
85	1	5	8	2	0.15	10, 95%	P2	<b>H1+H4+LS-RS</b>	6.699	5.87%	1.1	0.02
86	1	5	8	4	0.15	10, 95%	P2	<b>H1+H4+LS-RS</b>	5.227	11.99%	1.3	0.02
87	1	5	8	2	0.00	10, 95%	P2	<b>H1+H4+LS-RS</b>	7.317	7.64%	1.4	0.02
88	1	5	8	2	0.15	15, 95%	P2	<b>H1+H4+LS-RS</b>	6.699	5.87%	1.1	0.02
89	1	5	8	2	0.15	5, 95%	P2	<b>H1+H4+LS-RS</b>	7.105	4.99%	2.9	0.05
90	1	5	8	4	0.15	15, 95%	P2	<b>H1+H4+LS-RS</b>	4.945	5.92%	1.0	0.02
91	1	5	8	2	0.00	5, 95%	P2	<b>H1+H4+LS-RS</b>	8.149	5.93%	3.8	0.06
92	1	5	8	2	0.00	15, 95%	P2	<b>H1+H4+LS-RS</b>	7.317	7.64%	1.2	0.02
93	1	5	8	4	0.00	5, 95%	P2	<b>H1+H4+LS-RS</b>	6.405	2.42%	3.8	0.06
94	1	5	8	4	0.15	5, 95%	P2	<b>H1+H4+LS-RS</b>	5.703	7.76%	3.3	0.05
95	1	5	8	4	0.00	10, 95%	P2	<b>H1+H4+LS-RS</b>	5.820	8.33%	1.4	0.02
96	1	5	8	4	0.00	15, 95%	P2	<b>H1+H4+LS-RS</b>	5.820	8.33%	1.5	0.03
97	1	5	8	2	0.15	10, 95%	P3	<b>H1+H4+LS-RS</b>	5.607	1.74%	1.5	0.02

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
98	1	5	8	4	0.15	10, 95%	P3	<b>H1+H4+LS-RS</b>	3.421	0.00%	1.6	0.03
99	1	5	8	2	0.00	10, 95%	P3	<b>H1+H4+LS-RS</b>	6.586	2.17%	2.1	0.03
100	1	5	8	2	0.15	15, 95%	P3	<b>H1+H4+LS-RS</b>	5.607	1.74%	1.2	0.02
101	1	5	8	2	0.15	5, 95%	P3	<b>H1+H4+LS-RS</b>	6.510	3.20%	2.2	0.04
102	1	5	8	4	0.15	15, 95%	P3	<b>H1+H4+LS-RS</b>	3.421	0.00%	1.4	0.02
103	1	5	8	2	0.00	5, 95%	P3	<b>H1+H4+LS-RS</b>	7.542	1.23%	2.6	0.04
104	1	5	8	2	0.00	15, 95%	P3	<b>H1+H4+LS-RS</b>	6.586	2.17%	1.8	0.03
105	1	5	8	4	0.00	5, 95%	P3	<b>H1+H4+LS-RS</b>	5.200	5.53%	2.9	0.05
106	1	5	8	4	0.15	5, 95%	P3	<b>H1+H4+LS-RS</b>	3.928	2.25%	2.1	0.04
107	1	5	8	4	0.00	10, 95%	P3	<b>H1+H4+LS-RS</b>	4.402	0.00%	1.8	0.03
108	1	5	8	4	0.00	15, 95%	P3	<b>H1+H4+LS-RS</b>	4.402	0.00%	1.2	0.02
109	1	5	8	2	0.15	10, 95%	P1	<b>H1+H3+LS-GA</b>	6.007	0.65%	7.6	0.13
110	1	5	8	4	0.15	10, 95%	P1	<b>H1+H3+LS-GA</b>	4.362	0.00%	4.4	0.07
111	1	5	8	2	0.00	10, 95%	P1	<b>H1+H3+LS-GA</b>	6.914	0.34%	7.3	0.12
112	1	5	8	2	0.15	15, 95%	P1	<b>H1+H3+LS-GA</b>	5.639	0.00%	6.6	0.11
113	1	5	8	2	0.15	5, 95%	P1	<b>H1+H3+LS-GA</b>	7.010	7.18%	16.0	0.27
114	1	5	8	4	0.15	15, 95%	P1	<b>H1+H3+LS-GA</b>	4.191	5.70%	8.6	0.14
115	1	5	8	2	0.00	5, 95%	P1	<b>H1+H3+LS-GA</b>	7.739	2.23%	6.1	0.10
116	1	5	8	2	0.00	15, 95%	P1	<b>H1+H3+LS-GA</b>	6.536	0.91%	6.2	0.10
117	1	5	8	4	0.00	5, 95%	P1	<b>H1+H3+LS-GA</b>	5.752	3.49%	7.0	0.12
118	1	5	8	4	0.15	5, 95%	P1	<b>H1+H3+LS-GA</b>	4.663	2.53%	5.7	0.09
119	1	5	8	4	0.00	10, 95%	P1	<b>H1+H3+LS-GA</b>	4.915	0.00%	8.0	0.13
120	1	5	8	4	0.00	15, 95%	P1	<b>H1+H3+LS-GA</b>	4.761	2.04%	4.8	0.08
121	1	5	8	2	0.15	10, 95%	P2	<b>H1+H3+LS-GA</b>	6.328	0.00%	4.4	0.07
122	1	5	8	4	0.15	10, 95%	P2	<b>H1+H3+LS-GA</b>	4.669	0.02%	3.5	0.06
123	1	5	8	2	0.00	10, 95%	P2	<b>H1+H3+LS-GA</b>	6.798	0.00%	3.4	0.06
124	1	5	8	2	0.15	15, 95%	P2	<b>H1+H3+LS-GA</b>	6.328	0.00%	5.5	0.09
125	1	5	8	2	0.15	5, 95%	P2	<b>H1+H3+LS-GA</b>	7.105	4.99%	6.8	0.11
126	1	5	8	4	0.15	15, 95%	P2	<b>H1+H3+LS-GA</b>	4.669	0.00%	3.9	0.07
127	1	5	8	2	0.00	5, 95%	P2	<b>H1+H3+LS-GA</b>	7.788	1.25%	5.3	0.09
128	1	5	8	2	0.00	15, 95%	P2	<b>H1+H3+LS-GA</b>	6.798	0.00%	7.8	0.13
129	1	5	8	4	0.00	5, 95%	P2	<b>H1+H3+LS-GA</b>	6.405	2.42%	7.4	0.12
130	1	5	8	4	0.15	5, 95%	P2	<b>H1+H3+LS-GA</b>	5.550	4.87%	5.2	0.09
131	1	5	8	4	0.00	10, 95%	P2	<b>H1+H3+LS-GA</b>	5.455	1.54%	6.7	0.11
132	1	5	8	4	0.00	15, 95%	P2	<b>H1+H3+LS-GA</b>	5.455	1.54%	7.2	0.12
133	1	5	8	2	0.15	10, 95%	P3	<b>H1+H3+LS-GA</b>	5.607	1.74%	14.8	0.25
134	1	5	8	4	0.15	10, 95%	P3	<b>H1+H3+LS-GA</b>	3.421	0.00%	13.5	0.22
135	1	5	8	2	0.00	10, 95%	P3	<b>H1+H3+LS-GA</b>	6.586	2.17%	14.6	0.24
136	1	5	8	2	0.15	15, 95%	P3	<b>H1+H3+LS-GA</b>	5.607	1.74%	5.8	0.10
137	1	5	8	2	0.15	5, 95%	P3	<b>H1+H3+LS-GA</b>	6.308	0.00%	24.9	0.42
138	1	5	8	4	0.15	15, 95%	P3	<b>H1+H3+LS-GA</b>	3.421	0.00%	12.6	0.21
139	1	5	8	2	0.00	5, 95%	P3	<b>H1+H3+LS-GA</b>	7.542	1.23%	52.5	0.88
140	1	5	8	2	0.00	15, 95%	P3	<b>H1+H3+LS-GA</b>	6.592	2.26%	32.1	0.53
141	1	5	8	4	0.00	5, 95%	P3	<b>H1+H3+LS-GA</b>	5.200	5.53%	14.8	0.25
142	1	5	8	4	0.15	5, 95%	P3	<b>H1+H3+LS-GA</b>	3.928	2.25%	7.5	0.12
143	1	5	8	4	0.00	10, 95%	P3	<b>H1+H3+LS-GA</b>	4.402	0.00%	8.7	0.15
144	1	5	8	4	0.00	15, 95%	P3	<b>H1+H3+LS-GA</b>	4.402	0.00%	7.8	0.13
145	1	5	8	2	0.15	10, 95%	P1	<b>H1+H4+LS-GA</b>	6.007	0.65%	3.7	0.06
146	1	5	8	4	0.15	10, 95%	P1	<b>H1+H4+LS-GA</b>	4.362	0.00%	5.9	0.10
147	1	5	8	2	0.00	10, 95%	P1	<b>H1+H4+LS-GA</b>	6.914	0.34%	5.5	0.09
148	1	5	8	2	0.15	15, 95%	P1	<b>H1+H4+LS-GA</b>	5.639	0.00%	3.8	0.06
149	1	5	8	2	0.15	5, 95%	P1	<b>H1+H4+LS-GA</b>	6.801	3.99%	8.2	0.14
150	1	5	8	4	0.15	15, 95%	P1	<b>H1+H4+LS-GA</b>	3.966	0.00%	4.6	0.08
151	1	5	8	2	0.00	5, 95%	P1	<b>H1+H4+LS-GA</b>	7.739	2.23%	13.6	0.23
152	1	5	8	2	0.00	15, 95%	P1	<b>H1+H4+LS-GA</b>	6.536	0.91%	7.9	0.13
153	1	5	8	4	0.00	5, 95%	P1	<b>H1+H4+LS-GA</b>	6.166	10.94%	9.3	0.15
154	1	5	8	4	0.15	5, 95%	P1	<b>H1+H4+LS-GA</b>	4.663	2.53%	8.7	0.14
155	1	5	8	4	0.00	10, 95%	P1	<b>H1+H4+LS-GA</b>	4.915	0.00%	4.8	0.08
156	1	5	8	4	0.00	15, 95%	P1	<b>H1+H4+LS-GA</b>	4.761	2.04%	7.7	0.13
157	1	5	8	2	0.15	10, 95%	P2	<b>H1+H4+LS-GA</b>	6.328	0.00%	121.5	2.03
158	1	5	8	4	0.15	10, 95%	P2	<b>H1+H4+LS-GA</b>	5.092	9.08%	52.2	0.87
159	1	5	8	2	0.00	10, 95%	P2	<b>H1+H4+LS-GA</b>	7.317	7.64%	27.0	0.45
160	1	5	8	2	0.15	15, 95%	P2	<b>H1+H4+LS-GA</b>	6.699	5.87%	40.4	0.67
161	1	5	8	2	0.15	5, 95%	P2	<b>H1+H4+LS-GA</b>	7.105	4.99%	5.8	0.10

Table continues on next page



Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
162	1	5	8	4	0.15	15, 95%	P2	<b>H1+H4+LS-GA</b>	4.815	3.13%	12.3	0.20
163	1	5	8	2	0.00	5, 95%	P2	<b>H1+H4+LS-GA</b>	7.789	1.25%	13.3	0.22
164	1	5	8	2	0.00	15, 95%	P2	<b>H1+H4+LS-GA</b>	7.317	7.64%	40.9	0.68
165	1	5	8	4	0.00	5, 95%	P2	<b>H1+H4+LS-GA</b>	6.404	2.42%	5.9	0.10
166	1	5	8	4	0.15	5, 95%	P2	<b>H1+H4+LS-GA</b>	5.550	4.87%	38.4	0.64
167	1	5	8	4	0.00	10, 95%	P2	<b>H1+H4+LS-GA</b>	5.648	5.13%	21.3	0.35
168	1	5	8	4	0.00	15, 95%	P2	<b>H1+H4+LS-GA</b>	5.648	5.12%	466.5	7.77
169	1	5	8	2	0.15	10, 95%	P3	<b>H1+H4+LS-GA</b>	5.607	1.74%	11.8	0.20
170	1	5	8	4	0.15	10, 95%	P3	<b>H1+H4+LS-GA</b>	3.421	0.00%	5.7	0.09
171	1	5	8	2	0.00	10, 95%	P3	<b>H1+H4+LS-GA</b>	6.586	2.17%	24.7	0.41
172	1	5	8	2	0.15	15, 95%	P3	<b>H1+H4+LS-GA</b>	5.607	1.74%	44.0	0.73
173	1	5	8	2	0.15	5, 95%	P3	<b>H1+H4+LS-GA</b>	6.308	0.00%	13.6	0.23
174	1	5	8	4	0.15	15, 95%	P3	<b>H1+H4+LS-GA</b>	3.421	0.00%	8.6	0.14
175	1	5	8	2	0.00	5, 95%	P3	<b>H1+H4+LS-GA</b>	7.542	1.23%	43.5	0.72
176	1	5	8	2	0.00	15, 95%	P3	<b>H1+H4+LS-GA</b>	6.586	2.17%	12.1	0.20
177	1	5	8	4	0.00	5, 95%	P3	<b>H1+H4+LS-GA</b>	5.200	5.52%	14.3	0.24
178	1	5	8	4	0.15	5, 95%	P3	<b>H1+H4+LS-GA</b>	3.928	2.25%	10.5	0.18
179	1	5	8	4	0.00	10, 95%	P3	<b>H1+H4+LS-GA</b>	4.402	0.00%	14.0	0.23
180	1	5	8	4	0.00	15, 95%	P3	<b>H1+H4+LS-GA</b>	4.402	0.01%	10.4	0.17
181	1	5	8	2	0.15	10, 95%	P1	<b>H1+H3+LS-KK</b>	6.824	14.33%	2.7	0.05
182	1	5	8	4	0.15	10, 95%	P1	<b>H1+H3+LS-KK</b>	4.362	0.00%	2.2	0.04
183	1	5	8	2	0.00	10, 95%	P1	<b>H1+H3+LS-KK</b>	7.380	7.12%	2.0	0.03
184	1	5	8	2	0.15	15, 95%	P1	<b>H1+H3+LS-KK</b>	6.492	15.13%	1.5	0.03
185	1	5	8	2	0.15	5, 95%	P1	<b>H1+H3+LS-KK</b>	6.894	5.41%	3.2	0.05
186	1	5	8	4	0.15	15, 95%	P1	<b>H1+H3+LS-KK</b>	3.966	0.00%	1.0	0.02
187	1	5	8	2	0.00	5, 95%	P1	<b>H1+H3+LS-KK</b>	7.791	2.92%	4.9	0.08
188	1	5	8	2	0.00	15, 95%	P1	<b>H1+H3+LS-KK</b>	7.380	13.94%	1.9	0.03
189	1	5	8	4	0.00	5, 95%	P1	<b>H1+H3+LS-KK</b>	5.752	3.49%	4.1	0.07
190	1	5	8	4	0.15	5, 95%	P1	<b>H1+H3+LS-KK</b>	4.663	2.53%	2.7	0.04
191	1	5	8	4	0.00	10, 95%	P1	<b>H1+H3+LS-KK</b>	4.915	0.00%	2.4	0.04
192	1	5	8	4	0.00	15, 95%	P1	<b>H1+H3+LS-KK</b>	4.761	2.04%	1.2	0.02
193	1	5	8	2	0.15	10, 95%	P2	<b>H1+H3+LS-KK</b>	6.718	6.16%	0.9	0.02
194	1	5	8	4	0.15	10, 95%	P2	<b>H1+H3+LS-KK</b>	4.669	0.02%	0.7	0.01
195	1	5	8	2	0.00	10, 95%	P2	<b>H1+H3+LS-KK</b>	7.643	12.44%	1.2	0.02
196	1	5	8	2	0.15	15, 95%	P2	<b>H1+H3+LS-KK</b>	6.718	6.16%	0.9	0.01
197	1	5	8	2	0.15	5, 95%	P2	<b>H1+H3+LS-KK</b>	7.307	7.98%	2.7	0.05
198	1	5	8	4	0.15	15, 95%	P2	<b>H1+H3+LS-KK</b>	4.669	0.00%	0.8	0.01
199	1	5	8	2	0.00	5, 95%	P2	<b>H1+H3+LS-KK</b>	8.334	8.35%	3.5	0.06
200	1	5	8	2	0.00	15, 95%	P2	<b>H1+H3+LS-KK</b>	7.643	12.44%	1.2	0.02
201	1	5	8	4	0.00	5, 95%	P2	<b>H1+H3+LS-KK</b>	6.404	2.42%	2.7	0.05
202	1	5	8	4	0.15	5, 95%	P2	<b>H1+H3+LS-KK</b>	5.550	4.87%	2.2	0.04
203	1	5	8	4	0.00	10, 95%	P2	<b>H1+H3+LS-KK</b>	5.455	1.54%	1.0	0.02
204	1	5	8	4	0.00	15, 95%	P2	<b>H1+H3+LS-KK</b>	5.455	1.54%	0.9	0.02
205	1	5	8	2	0.15	10, 95%	P3	<b>H1+H3+LS-KK</b>	5.607	1.74%	1.0	0.02
206	1	5	8	4	0.15	10, 95%	P3	<b>H1+H3+LS-KK</b>	3.421	0.00%	0.6	0.01
207	1	5	8	2	0.00	10, 95%	P3	<b>H1+H3+LS-KK</b>	6.586	2.16%	1.2	0.02
208	1	5	8	2	0.15	15, 95%	P3	<b>H1+H3+LS-KK</b>	5.607	1.74%	0.9	0.01
209	1	5	8	2	0.15	5, 95%	P3	<b>H1+H3+LS-KK</b>	6.308	0.00%	2.3	0.04
210	1	5	8	4	0.15	15, 95%	P3	<b>H1+H3+LS-KK</b>	3.421	0.00%	0.6	0.01
211	1	5	8	2	0.00	5, 95%	P3	<b>H1+H3+LS-KK</b>	7.542	1.23%	3.3	0.06
212	1	5	8	2	0.00	15, 95%	P3	<b>H1+H3+LS-KK</b>	6.586	2.16%	1.1	0.02
213	1	5	8	4	0.00	5, 95%	P3	<b>H1+H3+LS-KK</b>	5.200	5.52%	3.2	0.05
214	1	5	8	4	0.15	5, 95%	P3	<b>H1+H3+LS-KK</b>	3.928	2.25%	1.6	0.03
215	1	5	8	4	0.00	10, 95%	P3	<b>H1+H3+LS-KK</b>	4.402	0.00%	1.0	0.02
216	1	5	8	4	0.00	15, 95%	P3	<b>H1+H3+LS-KK</b>	4.402	0.00%	0.9	0.02
217	1	5	8	2	0.15	10, 95%	P1	<b>H2+H3+LS-RS</b>	6.007	0.65%	2.1	0.03
218	1	5	8	4	0.15	10, 95%	P1	<b>H2+H3+LS-RS</b>	4.362	0.00%	2.5	0.04
219	1	5	8	2	0.00	10, 95%	P1	<b>H2+H3+LS-RS</b>	6.913	0.34%	4.2	0.07
220	1	5	8	2	0.15	15, 95%	P1	<b>H2+H3+LS-RS</b>	5.639	0.00%	1.7	0.03
221	1	5	8	2	0.15	5, 95%	P1	<b>H2+H3+LS-RS</b>	6.801	3.99%	5.3	0.09
222	1	5	8	4	0.15	15, 95%	P1	<b>H2+H3+LS-RS</b>	3.978	0.32%	1.2	0.02
223	1	5	8	2	0.00	5, 95%	P1	<b>H2+H3+LS-RS</b>	7.744	2.30%	5.5	0.09
224	1	5	8	2	0.00	15, 95%	P1	<b>H2+H3+LS-RS</b>	6.536	0.90%	1.5	0.02
225	1	5	8	4	0.00	5, 95%	P1	<b>H2+H3+LS-RS</b>	5.752	3.49%	4.1	0.07

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
226	1	5	8	4	0.15	5, 95%	P1	<b>H2+H3+LS-RS</b>	4.663	2.53%	2.8	0.05
227	1	5	8	4	0.00	10, 95%	P1	<b>H2+H3+LS-RS</b>	4.915	0.00%	2.6	0.04
228	1	5	8	4	0.00	15, 95%	P1	<b>H2+H3+LS-RS</b>	4.761	2.04%	1.5	0.02
229	1	5	8	2	0.15	10, 95%	P2	<b>H2+H3+LS-RS</b>	6.338	0.16%	1.2	0.02
230	1	5	8	4	0.15	10, 95%	P2	<b>H2+H3+LS-RS</b>	4.793	2.68%	1.0	0.02
231	1	5	8	2	0.00	10, 95%	P2	<b>H2+H3+LS-RS</b>	6.863	0.96%	1.4	0.02
232	1	5	8	2	0.15	15, 95%	P2	<b>H2+H3+LS-RS</b>	6.338	0.16%	1.4	0.02
233	1	5	8	2	0.15	5, 95%	P2	<b>H2+H3+LS-RS</b>	7.105	4.99%	4.7	0.08
234	1	5	8	4	0.15	15, 95%	P2	<b>H2+H3+LS-RS</b>	4.793	2.66%	1.0	0.02
235	1	5	8	2	0.00	5, 95%	P2	<b>H2+H3+LS-RS</b>	7.789	1.25%	2.6	0.04
236	1	5	8	2	0.00	15, 95%	P2	<b>H2+H3+LS-RS</b>	6.863	0.96%	1.4	0.02
237	1	5	8	4	0.00	5, 95%	P2	<b>H2+H3+LS-RS</b>	6.404	2.42%	3.9	0.07
238	1	5	8	4	0.15	5, 95%	P2	<b>H2+H3+LS-RS</b>	5.368	1.43%	3.3	0.05
239	1	5	8	4	0.00	10, 95%	P2	<b>H2+H3+LS-RS</b>	5.455	1.54%	1.2	0.02
240	1	5	8	4	0.00	15, 95%	P2	<b>H2+H3+LS-RS</b>	5.455	1.54%	1.3	0.02
241	1	5	8	2	0.15	10, 95%	P3	<b>H2+H3+LS-RS</b>	5.511	0.00%	0.9	0.02
242	1	5	8	4	0.15	10, 95%	P3	<b>H2+H3+LS-RS</b>	3.421	0.00%	0.9	0.01
243	1	5	8	2	0.00	10, 95%	P3	<b>H2+H3+LS-RS</b>	6.502	0.87%	1.5	0.03
244	1	5	8	2	0.15	15, 95%	P3	<b>H2+H3+LS-RS</b>	5.511	0.00%	1.1	0.02
245	1	5	8	2	0.15	5, 95%	P3	<b>H2+H3+LS-RS</b>	6.308	0.00%	2.7	0.05
246	1	5	8	4	0.15	15, 95%	P3	<b>H2+H3+LS-RS</b>	3.421	0.00%	1.3	0.02
247	1	5	8	2	0.00	5, 95%	P3	<b>H2+H3+LS-RS</b>	7.542	1.23%	2.8	0.05
248	1	5	8	2	0.00	15, 95%	P3	<b>H2+H3+LS-RS</b>	6.502	0.87%	1.4	0.02
249	1	5	8	4	0.00	5, 95%	P3	<b>H2+H3+LS-RS</b>	4.972	0.91%	2.7	0.04
250	1	5	8	4	0.15	5, 95%	P3	<b>H2+H3+LS-RS</b>	3.879	0.97%	1.6	0.03
251	1	5	8	4	0.00	10, 95%	P3	<b>H2+H3+LS-RS</b>	4.402	0.00%	1.8	0.03
252	1	5	8	4	0.00	15, 95%	P3	<b>H2+H3+LS-RS</b>	4.402	0.00%	1.3	0.02
253	1	5	8	2	0.15	10, 95%	P1	<b>H2+H4+LS-RS</b>	6.007	0.65%	2.0	0.03
254	1	5	8	4	0.15	10, 95%	P1	<b>H2+H4+LS-RS</b>	4.362	0.00%	2.8	0.05
255	1	5	8	2	0.00	10, 95%	P1	<b>H2+H4+LS-RS</b>	6.916	0.38%	3.3	0.06
256	1	5	8	2	0.15	15, 95%	P1	<b>H2+H4+LS-RS</b>	5.639	0.00%	2.0	0.03
257	1	5	8	2	0.15	5, 95%	P1	<b>H2+H4+LS-RS</b>	7.010	7.18%	6.0	0.10
258	1	5	8	4	0.15	15, 95%	P1	<b>H2+H4+LS-RS</b>	4.191	5.70%	1.7	0.03
259	1	5	8	2	0.00	5, 95%	P1	<b>H2+H4+LS-RS</b>	7.791	2.92%	3.8	0.06
260	1	5	8	2	0.00	15, 95%	P1	<b>H2+H4+LS-RS</b>	6.536	0.91%	2.1	0.04
261	1	5	8	4	0.00	5, 95%	P1	<b>H2+H4+LS-RS</b>	5.752	3.49%	3.5	0.06
262	1	5	8	4	0.15	5, 95%	P1	<b>H2+H4+LS-RS</b>	5.039	10.78%	5.0	0.08
263	1	5	8	4	0.00	10, 95%	P1	<b>H2+H4+LS-RS</b>	4.915	0.00%	2.4	0.04
264	1	5	8	4	0.00	15, 95%	P1	<b>H2+H4+LS-RS</b>	4.761	2.04%	1.6	0.03
265	1	5	8	2	0.15	10, 95%	P2	<b>H2+H4+LS-RS</b>	6.538	3.31%	1.6	0.03
266	1	5	8	4	0.15	10, 95%	P2	<b>H2+H4+LS-RS</b>	5.227	11.99%	1.3	0.02
267	1	5	8	2	0.00	10, 95%	P2	<b>H2+H4+LS-RS</b>	7.323	7.73%	5.4	0.09
268	1	5	8	2	0.15	15, 95%	P2	<b>H2+H4+LS-RS</b>	6.538	3.31%	1.4	0.02
269	1	5	8	2	0.15	5, 95%	P2	<b>H2+H4+LS-RS</b>	7.105	4.99%	3.5	0.06
270	1	5	8	4	0.15	15, 95%	P2	<b>H2+H4+LS-RS</b>	5.228	11.96%	1.1	0.02
271	1	5	8	2	0.00	5, 95%	P2	<b>H2+H4+LS-RS</b>	7.788	1.25%	3.7	0.06
272	1	5	8	2	0.00	15, 95%	P2	<b>H2+H4+LS-RS</b>	7.323	7.73%	1.3	0.02
273	1	5	8	4	0.00	5, 95%	P2	<b>H2+H4+LS-RS</b>	6.404	2.42%	4.1	0.07
274	1	5	8	4	0.15	5, 95%	P2	<b>H2+H4+LS-RS</b>	5.368	1.43%	2.7	0.05
275	1	5	8	4	0.00	10, 95%	P2	<b>H2+H4+LS-RS</b>	5.820	8.33%	1.4	0.02
276	1	5	8	4	0.00	15, 95%	P2	<b>H2+H4+LS-RS</b>	5.820	8.33%	1.1	0.02
277	1	5	8	2	0.15	10, 95%	P3	<b>H2+H4+LS-RS</b>	5.511	0.00%	1.2	0.02
278	1	5	8	4	0.15	10, 95%	P3	<b>H2+H4+LS-RS</b>	3.426	0.15%	1.2	0.02
279	1	5	8	2	0.00	10, 95%	P3	<b>H2+H4+LS-RS</b>	6.502	0.87%	1.5	0.03
280	1	5	8	2	0.15	15, 95%	P3	<b>H2+H4+LS-RS</b>	5.511	0.00%	1.3	0.02
281	1	5	8	2	0.15	5, 95%	P3	<b>H2+H4+LS-RS</b>	6.308	0.00%	2.3	0.04
282	1	5	8	4	0.15	15, 95%	P3	<b>H2+H4+LS-RS</b>	3.421	0.00%	2.1	0.03
283	1	5	8	2	0.00	5, 95%	P3	<b>H2+H4+LS-RS</b>	7.542	1.23%	2.4	0.04
284	1	5	8	2	0.00	15, 95%	P3	<b>H2+H4+LS-RS</b>	6.502	0.87%	1.8	0.03
285	1	5	8	4	0.00	5, 95%	P3	<b>H2+H4+LS-RS</b>	4.973	0.91%	2.2	0.04
286	1	5	8	4	0.15	5, 95%	P3	<b>H2+H4+LS-RS</b>	3.879	0.98%	1.7	0.03
287	1	5	8	4	0.00	10, 95%	P3	<b>H2+H4+LS-RS</b>	4.402	0.00%	1.1	0.02
288	1	5	8	4	0.00	15, 95%	P3	<b>H2+H4+LS-RS</b>	4.402	0.00%	5.0	0.08
289	1	5	8	2	0.15	10, 95%	P1	<b>H2+H3+LS-GA</b>	6.007	0.65%	5.0	0.08

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
290	1	5	8	4	0.15	10, 95%	P1	<b>H2+H3+LS-GA</b>	4.362	0.00%	8.7	0.14
291	1	5	8	2	0.00	10, 95%	P1	<b>H2+H3+LS-GA</b>	6.914	0.34%	5.5	0.09
292	1	5	8	2	0.15	15, 95%	P1	<b>H2+H3+LS-GA</b>	5.639	0.00%	6.2	0.10
293	1	5	8	2	0.15	5, 95%	P1	<b>H2+H3+LS-GA</b>	6.801	3.99%	9.2	0.15
294	1	5	8	4	0.15	15, 95%	P1	<b>H2+H3+LS-GA</b>	3.966	0.00%	3.7	0.06
295	1	5	8	2	0.00	5, 95%	P1	<b>H2+H3+LS-GA</b>	7.739	2.23%	13.1	0.22
296	1	5	8	2	0.00	15, 95%	P1	<b>H2+H3+LS-GA</b>	6.536	0.91%	4.2	0.07
297	1	5	8	4	0.00	5, 95%	P1	<b>H2+H3+LS-GA</b>	5.752	3.49%	7.1	0.12
298	1	5	8	4	0.15	5, 95%	P1	<b>H2+H3+LS-GA</b>	4.663	2.53%	5.1	0.08
299	1	5	8	4	0.00	10, 95%	P1	<b>H2+H3+LS-GA</b>	4.915	0.00%	5.0	0.08
300	1	5	8	4	0.00	15, 95%	P1	<b>H2+H3+LS-GA</b>	4.761	2.04%	4.8	0.08
301	1	5	8	2	0.15	10, 95%	P2	<b>H2+H3+LS-GA</b>	6.338	0.16%	4.2	0.07
302	1	5	8	4	0.15	10, 95%	P2	<b>H2+H3+LS-GA</b>	4.793	2.68%	3.6	0.06
303	1	5	8	2	0.00	10, 95%	P2	<b>H2+H3+LS-GA</b>	6.863	0.96%	4.7	0.08
304	1	5	8	2	0.15	15, 95%	P2	<b>H2+H3+LS-GA</b>	6.338	0.16%	4.5	0.07
305	1	5	8	2	0.15	5, 95%	P2	<b>H2+H3+LS-GA</b>	7.105	4.99%	8.1	0.14
306	1	5	8	4	0.15	15, 95%	P2	<b>H2+H3+LS-GA</b>	4.793	2.65%	3.9	0.07
307	1	5	8	2	0.00	5, 95%	P2	<b>H2+H3+LS-GA</b>	7.789	1.25%	7.0	0.12
308	1	5	8	2	0.00	15, 95%	P2	<b>H2+H3+LS-GA</b>	6.863	0.96%	4.5	0.07
309	1	5	8	4	0.00	5, 95%	P2	<b>H2+H3+LS-GA</b>	6.404	2.42%	8.9	0.15
310	1	5	8	4	0.15	5, 95%	P2	<b>H2+H3+LS-GA</b>	5.368	1.43%	6.3	0.10
311	1	5	8	4	0.00	10, 95%	P2	<b>H2+H3+LS-GA</b>	5.455	1.54%	6.4	0.11
312	1	5	8	4	0.00	15, 95%	P2	<b>H2+H3+LS-GA</b>	5.455	1.54%	6.3	0.10
313	1	5	8	2	0.15	10, 95%	P3	<b>H2+H3+LS-GA</b>	5.511	0.00%	5.3	0.09
314	1	5	8	4	0.15	10, 95%	P3	<b>H2+H3+LS-GA</b>	3.421	0.00%	9.2	0.15
315	1	5	8	2	0.00	10, 95%	P3	<b>H2+H3+LS-GA</b>	6.509	0.98%	68.3	1.14
316	1	5	8	2	0.15	15, 95%	P3	<b>H2+H3+LS-GA</b>	5.511	0.00%	5.3	0.09
317	1	5	8	2	0.15	5, 95%	P3	<b>H2+H3+LS-GA</b>	6.308	0.00%	12.3	0.21
318	1	5	8	4	0.15	15, 95%	P3	<b>H2+H3+LS-GA</b>	3.421	0.00%	7.1	0.12
319	1	5	8	2	0.00	5, 95%	P3	<b>H2+H3+LS-GA</b>	7.542	1.23%	36.2	0.60
320	1	5	8	2	0.00	15, 95%	P3	<b>H2+H3+LS-GA</b>	6.502	0.87%	56.0	0.93
321	1	5	8	4	0.00	5, 95%	P3	<b>H2+H3+LS-GA</b>	4.973	0.91%	10.6	0.18
322	1	5	8	4	0.15	5, 95%	P3	<b>H2+H3+LS-GA</b>	3.872	0.80%	7.0	0.12
323	1	5	8	4	0.00	10, 95%	P3	<b>H2+H3+LS-GA</b>	4.402	0.00%	7.9	0.13
324	1	5	8	4	0.00	15, 95%	P3	<b>H2+H3+LS-GA</b>	4.402	0.00%	7.3	0.12
325	1	5	8	2	0.15	10, 95%	P1	<b>H2+H4+LS-GA</b>	6.007	0.65%	6.1	0.10
326	1	5	8	4	0.15	10, 95%	P1	<b>H2+H4+LS-GA</b>	4.362	0.00%	5.7	0.10
327	1	5	8	2	0.00	10, 95%	P1	<b>H2+H4+LS-GA</b>	6.914	0.34%	19.2	0.32
328	1	5	8	2	0.15	15, 95%	P1	<b>H2+H4+LS-GA</b>	5.639	0.00%	3.7	0.06
329	1	5	8	2	0.15	5, 95%	P1	<b>H2+H4+LS-GA</b>	6.801	3.99%	6.1	0.10
330	1	5	8	4	0.15	15, 95%	P1	<b>H2+H4+LS-GA</b>	3.966	0.00%	3.6	0.06
331	1	5	8	2	0.00	5, 95%	P1	<b>H2+H4+LS-GA</b>	8.243	8.88%	62.4	1.04
332	1	5	8	2	0.00	15, 95%	P1	<b>H2+H4+LS-GA</b>	6.536	0.91%	6.7	0.11
333	1	5	8	4	0.00	5, 95%	P1	<b>H2+H4+LS-GA</b>	5.752	3.49%	6.6	0.11
334	1	5	8	4	0.15	5, 95%	P1	<b>H2+H4+LS-GA</b>	4.663	2.53%	5.3	0.09
335	1	5	8	4	0.00	10, 95%	P1	<b>H2+H4+LS-GA</b>	4.915	0.00%	4.9	0.08
336	1	5	8	4	0.00	15, 95%	P1	<b>H2+H4+LS-GA</b>	4.761	2.04%	26.3	0.44
337	1	5	8	2	0.15	10, 95%	P2	<b>H2+H4+LS-GA</b>	6.538	3.31%	7.5	0.13
338	1	5	8	4	0.15	10, 95%	P2	<b>H2+H4+LS-GA</b>	4.941	5.85%	39.9	0.66
339	1	5	8	2	0.00	10, 95%	P2	<b>H2+H4+LS-GA</b>	7.323	7.73%	4.8	0.08
340	1	5	8	2	0.15	15, 95%	P2	<b>H2+H4+LS-GA</b>	6.338	0.16%	27.0	0.45
341	1	5	8	2	0.15	5, 95%	P2	<b>H2+H4+LS-GA</b>	7.105	4.99%	9.7	0.16
342	1	5	8	4	0.15	15, 95%	P2	<b>H2+H4+LS-GA</b>	4.940	5.80%	94.9	1.58
343	1	5	8	2	0.00	5, 95%	P2	<b>H2+H4+LS-GA</b>	7.789	1.25%	7.2	0.12
344	1	5	8	2	0.00	15, 95%	P2	<b>H2+H4+LS-GA</b>	7.323	7.73%	4.5	0.08
345	1	5	8	4	0.00	5, 95%	P2	<b>H2+H4+LS-GA</b>	6.405	2.42%	15.5	0.26
346	1	5	8	4	0.15	5, 95%	P2	<b>H2+H4+LS-GA</b>	5.368	1.43%	9.0	0.15
347	1	5	8	4	0.00	10, 95%	P2	<b>H2+H4+LS-GA</b>	5.820	8.33%	440.8	7.35
348	1	5	8	4	0.00	15, 95%	P2	<b>H2+H4+LS-GA</b>	5.455	1.54%	379.6	6.33
349	1	5	8	2	0.15	10, 95%	P3	<b>H2+H4+LS-GA</b>	5.511	0.00%	7.8	0.13
350	1	5	8	4	0.15	10, 95%	P3	<b>H2+H4+LS-GA</b>	3.421	0.00%	13.8	0.23
351	1	5	8	2	0.00	10, 95%	P3	<b>H2+H4+LS-GA</b>	6.509	0.98%	20.1	0.34
352	1	5	8	2	0.15	15, 95%	P3	<b>H2+H4+LS-GA</b>	5.511	0.00%	17.6	0.29
353	1	5	8	2	0.15	5, 95%	P3	<b>H2+H4+LS-GA</b>	6.308	0.00%	16.1	0.27

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
354	1	5	8	4	0.15	15, 95%	P3	<b>H2+H4+LS-GA</b>	3.421	0.00%	4.9	0.08
355	1	5	8	2	0.00	5, 95%	P3	<b>H2+H4+LS-GA</b>	7.542	1.23%	10.1	0.17
356	1	5	8	2	0.00	15, 95%	P3	<b>H2+H4+LS-GA</b>	6.509	0.98%	32.5	0.54
357	1	5	8	4	0.00	5, 95%	P3	<b>H2+H4+LS-GA</b>	4.972	0.91%	9.6	0.16
358	1	5	8	4	0.15	5, 95%	P3	<b>H2+H4+LS-GA</b>	3.873	0.80%	17.7	0.29
359	1	5	8	4	0.00	10, 95%	P3	<b>H2+H4+LS-GA</b>	4.402	0.00%	8.7	0.15
360	1	5	8	4	0.00	15, 95%	P3	<b>H2+H4+LS-GA</b>	4.402	0.00%	10.2	0.17
361	1	5	8	2	0.15	10, 95%	P1	<b>H2+H3+LS-KK</b>	6.824	14.33%	1.2	0.02
362	1	5	8	4	0.15	10, 95%	P1	<b>H2+H3+LS-KK</b>	4.362	0.00%	0.8	0.01
363	1	5	8	2	0.00	10, 95%	P1	<b>H2+H3+LS-KK</b>	7.380	7.12%	1.4	0.02
364	1	5	8	2	0.15	15, 95%	P1	<b>H2+H3+LS-KK</b>	6.492	15.13%	1.0	0.02
365	1	5	8	2	0.15	5, 95%	P1	<b>H2+H3+LS-KK</b>	6.894	5.41%	3.2	0.05
366	1	5	8	4	0.15	15, 95%	P1	<b>H2+H3+LS-KK</b>	3.966	0.00%	1.1	0.02
367	1	5	8	2	0.00	5, 95%	P1	<b>H2+H3+LS-KK</b>	7.791	2.92%	4.9	0.08
368	1	5	8	2	0.00	15, 95%	P1	<b>H2+H3+LS-KK</b>	7.380	13.94%	1.9	0.03
369	1	5	8	4	0.00	5, 95%	P1	<b>H2+H3+LS-KK</b>	5.752	3.49%	3.9	0.07
370	1	5	8	4	0.15	5, 95%	P1	<b>H2+H3+LS-KK</b>	4.663	2.53%	2.7	0.04
371	1	5	8	4	0.00	10, 95%	P1	<b>H2+H3+LS-KK</b>	4.915	0.00%	2.8	0.05
372	1	5	8	4	0.00	15, 95%	P1	<b>H2+H3+LS-KK</b>	4.761	2.04%	1.2	0.02
373	1	5	8	2	0.15	10, 95%	P2	<b>H2+H3+LS-KK</b>	6.880	8.72%	1.5	0.02
374	1	5	8	4	0.15	10, 95%	P2	<b>H2+H3+LS-KK</b>	4.793	2.68%	0.9	0.01
375	1	5	8	2	0.00	10, 95%	P2	<b>H2+H3+LS-KK</b>	7.495	10.27%	1.4	0.02
376	1	5	8	2	0.15	15, 95%	P2	<b>H2+H3+LS-KK</b>	6.880	8.72%	1.3	0.02
377	1	5	8	2	0.15	5, 95%	P2	<b>H2+H3+LS-KK</b>	7.307	7.98%	4.0	0.07
378	1	5	8	4	0.15	15, 95%	P2	<b>H2+H3+LS-KK</b>	4.793	2.65%	0.8	0.01
379	1	5	8	2	0.00	5, 95%	P2	<b>H2+H3+LS-KK</b>	8.334	8.35%	3.5	0.06
380	1	5	8	2	0.00	15, 95%	P2	<b>H2+H3+LS-KK</b>	7.495	10.27%	1.2	0.02
381	1	5	8	4	0.00	5, 95%	P2	<b>H2+H3+LS-KK</b>	6.404	2.42%	3.5	0.06
382	1	5	8	4	0.15	5, 95%	P2	<b>H2+H3+LS-KK</b>	5.368	1.43%	2.3	0.04
383	1	5	8	4	0.00	10, 95%	P2	<b>H2+H3+LS-KK</b>	5.455	1.54%	1.0	0.02
384	1	5	8	4	0.00	15, 95%	P2	<b>H2+H3+LS-KK</b>	5.455	1.54%	1.0	0.02
385	1	5	8	2	0.15	10, 95%	P3	<b>H2+H3+LS-KK</b>	5.511	0.00%	0.8	0.01
386	1	5	8	4	0.15	10, 95%	P3	<b>H2+H3+LS-KK</b>	3.421	0.00%	0.6	0.01
387	1	5	8	2	0.00	10, 95%	P3	<b>H2+H3+LS-KK</b>	6.502	0.87%	1.2	0.02
388	1	5	8	2	0.15	15, 95%	P3	<b>H2+H3+LS-KK</b>	5.511	0.00%	0.8	0.01
389	1	5	8	2	0.15	5, 95%	P3	<b>H2+H3+LS-KK</b>	6.308	0.00%	2.7	0.05
390	1	5	8	4	0.15	15, 95%	P3	<b>H2+H3+LS-KK</b>	3.421	0.00%	0.6	0.01
391	1	5	8	2	0.00	5, 95%	P3	<b>H2+H3+LS-KK</b>	7.542	1.23%	2.9	0.05
392	1	5	8	2	0.00	15, 95%	P3	<b>H2+H3+LS-KK</b>	6.502	0.87%	1.1	0.02
393	1	5	8	4	0.00	5, 95%	P3	<b>H2+H3+LS-KK</b>	4.972	0.91%	2.2	0.04
394	1	5	8	4	0.15	5, 95%	P3	<b>H2+H3+LS-KK</b>	3.872	0.80%	1.5	0.02
395	1	5	8	4	0.00	10, 95%	P3	<b>H2+H3+LS-KK</b>	4.402	0.00%	1.0	0.02
396	1	5	8	4	0.00	15, 95%	P3	<b>H2+H3+LS-KK</b>	4.402	0.00%	0.9	0.02
397	1	5	8	2	0.15	10, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	5.969	0.00%	43.9	0.73
398	1	5	8	4	0.15	10, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	4.362	0.00%	46.8	0.78
399	1	5	8	2	0.00	10, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	6.892	0.02%	63.3	1.05
400	1	5	8	2	0.15	15, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	5.639	0.00%	38.3	0.64
401	1	5	8	2	0.15	5, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	6.591	0.79%	63.4	1.06
402	1	5	8	4	0.15	15, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	4.191	5.70%	39.4	0.66
403	1	5	8	2	0.00	5, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	7.575	0.06%	84.2	1.40
404	1	5	8	2	0.00	15, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	6.477	0.00%	51.2	0.85
405	1	5	8	4	0.00	5, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	5.567	0.16%	60.0	1.00
406	1	5	8	4	0.15	5, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	4.548	0.00%	56.3	0.94
407	1	5	8	4	0.00	10, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	4.915	0.00%	42.4	0.71
408	1	5	8	4	0.00	15, 95%	P1	<b>H1+H3+LS-CC+LS-RS</b>	4.761	2.04%	48.0	0.80
409	1	5	8	2	0.15	10, 95%	P2	<b>H1+H3+LS-CC+LS-RS</b>	6.328	0.00%	28.4	0.47
410	1	5	8	4	0.15	10, 95%	P2	<b>H1+H3+LS-CC+LS-RS</b>	4.669	0.02%	26.0	0.43
411	1	5	8	2	0.00	10, 95%	P2	<b>H1+H3+LS-CC+LS-RS</b>	6.797	0.00%	25.8	0.43
412	1	5	8	2	0.15	15, 95%	P2	<b>H1+H3+LS-CC+LS-RS</b>	6.328	0.00%	27.7	0.46
413	1	5	8	2	0.15	5, 95%	P2	<b>H1+H3+LS-CC+LS-RS</b>	7.105	4.99%	43.8	0.73
414	1	5	8	4	0.15	15, 95%	P2	<b>H1+H3+LS-CC+LS-RS</b>	4.669	0.00%	21.2	0.35
415	1	5	8	2	0.00	5, 95%	P2	<b>H1+H3+LS-CC+LS-RS</b>	7.788	1.25%	39.9	0.66
416	1	5	8	2	0.00	15, 95%	P2	<b>H1+H3+LS-CC+LS-RS</b>	6.798	0.00%	26.8	0.45
417	1	5	8	4	0.00	5, 95%	P2	<b>H1+H3+LS-CC+LS-RS</b>	6.253	0.00%	51.8	0.86

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
418	1	5	8	4	0.15	5, 95%	P2	H1+H3+LS-CC+LS-RS	5.292	0.00%	34.7	0.58
419	1	5	8	4	0.00	10, 95%	P2	H1+H3+LS-CC+LS-RS	5.373	0.00%	44.3	0.74
420	1	5	8	4	0.00	15, 95%	P2	H1+H3+LS-CC+LS-RS	5.373	0.00%	27.8	0.46
421	1	5	8	2	0.15	10, 95%	P3	H1+H3+LS-CC+LS-RS	5.540	0.53%	57.7	0.96
422	1	5	8	4	0.15	10, 95%	P3	H1+H3+LS-CC+LS-RS	3.421	0.00%	60.7	1.01
423	1	5	8	2	0.00	10, 95%	P3	H1+H3+LS-CC+LS-RS	6.483	0.58%	61.3	1.02
424	1	5	8	2	0.15	15, 95%	P3	H1+H3+LS-CC+LS-RS	5.540	0.53%	56.1	0.93
425	1	5	8	2	0.15	5, 95%	P3	H1+H3+LS-CC+LS-RS	6.308	0.00%	76.2	1.27
426	1	5	8	4	0.15	15, 95%	P3	H1+H3+LS-CC+LS-RS	3.440	0.57%	38.5	0.64
427	1	5	8	2	0.00	5, 95%	P3	H1+H3+LS-CC+LS-RS	7.450	0.00%	79.1	1.32
428	1	5	8	2	0.00	15, 95%	P3	H1+H3+LS-CC+LS-RS	6.484	0.58%	86.4	1.44
429	1	5	8	4	0.00	5, 95%	P3	H1+H3+LS-CC+LS-RS	5.156	4.64%	108.3	1.81
430	1	5	8	4	0.15	5, 95%	P3	H1+H3+LS-CC+LS-RS	3.872	0.80%	44.3	0.74
431	1	5	8	4	0.00	10, 95%	P3	H1+H3+LS-CC+LS-RS	4.402	0.00%	59.2	0.99
432	1	5	8	4	0.00	15, 95%	P3	H1+H3+LS-CC+LS-RS	4.402	0.00%	61.4	1.02
433	1	5	8	2	0.15	10, 95%	P1	H1+H4+LS-CC+LS-RS	5.969	0.00%	38.9	0.65
434	1	5	8	4	0.15	10, 95%	P1	H1+H4+LS-CC+LS-RS	4.452	2.08%	37.5	0.62
435	1	5	8	2	0.00	10, 95%	P1	H1+H4+LS-CC+LS-RS	6.890	0.00%	82.7	1.38
436	1	5	8	2	0.15	15, 95%	P1	H1+H4+LS-CC+LS-RS	5.639	0.00%	39.4	0.66
437	1	5	8	2	0.15	5, 95%	P1	H1+H4+LS-CC+LS-RS	6.592	0.79%	62.4	1.04
438	1	5	8	4	0.15	15, 95%	P1	H1+H4+LS-CC+LS-RS	4.191	5.70%	50.1	0.84
439	1	5	8	2	0.00	5, 95%	P1	H1+H4+LS-CC+LS-RS	7.570	0.00%	97.9	1.63
440	1	5	8	2	0.00	15, 95%	P1	H1+H4+LS-CC+LS-RS	6.478	0.02%	47.8	0.80
441	1	5	8	4	0.00	5, 95%	P1	H1+H4+LS-CC+LS-RS	5.567	0.16%	61.1	1.02
442	1	5	8	4	0.15	5, 95%	P1	H1+H4+LS-CC+LS-RS	4.548	0.00%	45.9	0.76
443	1	5	8	4	0.00	10, 95%	P1	H1+H4+LS-CC+LS-RS	4.915	0.00%	48.0	0.80
444	1	5	8	4	0.00	15, 95%	P1	H1+H4+LS-CC+LS-RS	4.761	2.04%	40.0	0.67
445	1	5	8	2	0.15	10, 95%	P2	H1+H4+LS-CC+LS-RS	6.538	3.31%	39.8	0.66
446	1	5	8	4	0.15	10, 95%	P2	H1+H4+LS-CC+LS-RS	5.014	7.42%	26.8	0.45
447	1	5	8	2	0.00	10, 95%	P2	H1+H4+LS-CC+LS-RS	7.317	7.64%	29.9	0.50
448	1	5	8	2	0.15	15, 95%	P2	H1+H4+LS-CC+LS-RS	6.538	3.31%	26.0	0.43
449	1	5	8	2	0.15	5, 95%	P2	H1+H4+LS-CC+LS-RS	7.105	4.99%	31.6	0.53
450	1	5	8	4	0.15	15, 95%	P2	H1+H4+LS-CC+LS-RS	4.827	3.38%	24.4	0.41
451	1	5	8	2	0.00	5, 95%	P2	H1+H4+LS-CC+LS-RS	7.789	1.25%	36.9	0.61
452	1	5	8	2	0.00	15, 95%	P2	H1+H4+LS-CC+LS-RS	7.317	7.64%	31.8	0.53
453	1	5	8	4	0.00	5, 95%	P2	H1+H4+LS-CC+LS-RS	6.253	0.00%	33.2	0.55
454	1	5	8	4	0.15	5, 95%	P2	H1+H4+LS-CC+LS-RS	5.368	1.43%	26.1	0.43
455	1	5	8	4	0.00	10, 95%	P2	H1+H4+LS-CC+LS-RS	5.553	3.36%	31.5	0.53
456	1	5	8	4	0.00	15, 95%	P2	H1+H4+LS-CC+LS-RS	5.609	4.40%	26.6	0.44
457	1	5	8	2	0.15	10, 95%	P3	H1+H4+LS-CC+LS-RS	5.540	0.53%	60.7	1.01
458	1	5	8	4	0.15	10, 95%	P3	H1+H4+LS-CC+LS-RS	3.421	0.00%	36.6	0.61
459	1	5	8	2	0.00	10, 95%	P3	H1+H4+LS-CC+LS-RS	6.484	0.58%	71.0	1.18
460	1	5	8	2	0.15	15, 95%	P3	H1+H4+LS-CC+LS-RS	5.540	0.53%	38.6	0.64
461	1	5	8	2	0.15	5, 95%	P3	H1+H4+LS-CC+LS-RS	6.308	0.00%	68.9	1.15
462	1	5	8	4	0.15	15, 95%	P3	H1+H4+LS-CC+LS-RS	3.421	0.00%	52.2	0.87
463	1	5	8	2	0.00	5, 95%	P3	H1+H4+LS-CC+LS-RS	7.450	0.00%	83.5	1.39
464	1	5	8	2	0.00	15, 95%	P3	H1+H4+LS-CC+LS-RS	6.484	0.58%	76.3	1.27
465	1	5	8	4	0.00	5, 95%	P3	H1+H4+LS-CC+LS-RS	4.957	0.59%	70.0	1.17
466	1	5	8	4	0.15	5, 95%	P3	H1+H4+LS-CC+LS-RS	3.881	1.01%	47.4	0.79
467	1	5	8	4	0.00	10, 95%	P3	H1+H4+LS-CC+LS-RS	4.402	0.00%	55.4	0.92
468	1	5	8	4	0.00	15, 95%	P3	H1+H4+LS-CC+LS-RS	4.402	0.00%	70.5	1.17
469	1	5	8	2	0.15	10, 95%	P1	H1+H3+LS-CC+LS-GA	5.969	0.00%	180.5	3.01
470	1	5	8	4	0.15	10, 95%	P1	H1+H3+LS-CC+LS-GA	4.362	0.00%	72.4	1.21
471	1	5	8	2	0.00	10, 95%	P1	H1+H3+LS-CC+LS-GA	6.890	0.00%	165.5	2.76
472	1	5	8	2	0.15	15, 95%	P1	H1+H3+LS-CC+LS-GA	5.639	0.00%	123.7	2.06
473	1	5	8	2	0.15	5, 95%	P1	H1+H3+LS-CC+LS-GA	6.591	0.79%	112.3	1.87
474	1	5	8	4	0.15	15, 95%	P1	H1+H3+LS-CC+LS-GA	3.966	0.00%	93.1	1.55
475	1	5	8	2	0.00	5, 95%	P1	H1+H3+LS-CC+LS-GA	7.570	0.00%	222.8	3.71
476	1	5	8	2	0.00	15, 95%	P1	H1+H3+LS-CC+LS-GA	6.477	0.00%	107.9	1.80
477	1	5	8	4	0.00	5, 95%	P1	H1+H3+LS-CC+LS-GA	5.567	0.16%	153.7	2.56
478	1	5	8	4	0.15	5, 95%	P1	H1+H3+LS-CC+LS-GA	4.548	0.00%	236.4	3.94
479	1	5	8	4	0.00	10, 95%	P1	H1+H3+LS-CC+LS-GA	4.915	0.00%	99.9	1.67
480	1	5	8	4	0.00	15, 95%	P1	H1+H3+LS-CC+LS-GA	4.761	2.04%	107.9	1.80
481	1	5	8	2	0.15	10, 95%	P2	H1+H3+LS-CC+LS-GA	6.328	0.00%	70.6	1.18

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
482	1	5	8	4	0.15	10, 95%	P2	H1+H3+LS-CC+LS-GA	4.669	0.02%	121.5	2.02
483	1	5	8	2	0.00	10, 95%	P2	H1+H3+LS-CC+LS-GA	6.798	0.00%	75.2	1.25
484	1	5	8	2	0.15	15, 95%	P2	H1+H3+LS-CC+LS-GA	6.328	0.00%	73.2	1.22
485	1	5	8	2	0.15	5, 95%	P2	H1+H3+LS-CC+LS-GA	7.105	4.99%	102.5	1.71
486	1	5	8	4	0.15	15, 95%	P2	H1+H3+LS-CC+LS-GA	4.669	0.00%	86.0	1.43
487	1	5	8	2	0.00	5, 95%	P2	H1+H3+LS-CC+LS-GA	7.788	1.25%	83.9	1.40
488	1	5	8	2	0.00	15, 95%	P2	H1+H3+LS-CC+LS-GA	6.798	0.00%	73.8	1.23
489	1	5	8	4	0.00	5, 95%	P2	H1+H3+LS-CC+LS-GA	6.253	0.00%	117.0	1.95
490	1	5	8	4	0.15	5, 95%	P2	H1+H3+LS-CC+LS-GA	5.292	0.00%	95.5	1.59
491	1	5	8	4	0.00	10, 95%	P2	H1+H3+LS-CC+LS-GA	5.373	0.00%	129.0	2.15
492	1	5	8	4	0.00	15, 95%	P2	H1+H3+LS-CC+LS-GA	5.373	0.00%	89.4	1.49
493	1	5	8	2	0.15	10, 95%	P3	H1+H3+LS-CC+LS-GA	5.511	0.00%	522.7	8.71
494	1	5	8	4	0.15	10, 95%	P3	H1+H3+LS-CC+LS-GA	3.421	0.00%	288.6	4.81
495	1	5	8	2	0.00	10, 95%	P3	H1+H3+LS-CC+LS-GA	6.484	0.58%	490.3	8.17
496	1	5	8	2	0.15	15, 95%	P3	H1+H3+LS-CC+LS-GA	5.511	0.00%	447.5	7.46
497	1	5	8	2	0.15	5, 95%	P3	H1+H3+LS-CC+LS-GA	6.308	0.00%	644.1	10.74
498	1	5	8	4	0.15	15, 95%	P3	H1+H3+LS-CC+LS-GA	3.421	0.00%	195.4	3.26
499	1	5	8	2	0.00	5, 95%	P3	H1+H3+LS-CC+LS-GA	7.450	0.00%	442.2	7.37
500	1	5	8	2	0.00	15, 95%	P3	H1+H3+LS-CC+LS-GA	6.484	0.58%	297.0	4.95
501	1	5	8	4	0.00	5, 95%	P3	H1+H3+LS-CC+LS-GA	4.955	0.56%	354.9	5.91
502	1	5	8	4	0.15	5, 95%	P3	H1+H3+LS-CC+LS-GA	3.872	0.80%	246.2	4.10
503	1	5	8	4	0.00	10, 95%	P3	H1+H3+LS-CC+LS-GA	4.402	0.00%	408.8	6.81
504	1	5	8	4	0.00	15, 95%	P3	H1+H3+LS-CC+LS-GA	4.402	0.00%	1766.7	29.45
505	1	5	8	2	0.15	10, 95%	P1	H1+H4+LS-CC+LS-GA	5.969	0.00%	340.4	5.67
506	1	5	8	4	0.15	10, 95%	P1	H1+H4+LS-CC+LS-GA	4.362	0.00%	82.7	1.38
507	1	5	8	2	0.00	10, 95%	P1	H1+H4+LS-CC+LS-GA	6.890	0.00%	197.0	3.28
508	1	5	8	2	0.15	15, 95%	P1	H1+H4+LS-CC+LS-GA	5.639	0.00%	73.6	1.23
509	1	5	8	2	0.15	5, 95%	P1	H1+H4+LS-CC+LS-GA	6.591	0.79%	114.6	1.91
510	1	5	8	4	0.15	15, 95%	P1	H1+H4+LS-CC+LS-GA	4.191	5.70%	254.1	4.23
511	1	5	8	2	0.00	5, 95%	P1	H1+H4+LS-CC+LS-GA	7.570	0.00%	755.7	12.59
512	1	5	8	2	0.00	15, 95%	P1	H1+H4+LS-CC+LS-GA	6.477	0.00%	309.6	5.16
513	1	5	8	4	0.00	5, 95%	P1	H1+H4+LS-CC+LS-GA	5.567	0.16%	177.6	2.96
514	1	5	8	4	0.15	5, 95%	P1	H1+H4+LS-CC+LS-GA	4.548	0.00%	123.9	2.06
515	1	5	8	4	0.00	10, 95%	P1	H1+H4+LS-CC+LS-GA	4.915	0.00%	153.0	2.55
516	1	5	8	4	0.00	15, 95%	P1	H1+H4+LS-CC+LS-GA	4.761	2.04%	109.5	1.83
517	1	5	8	2	0.15	10, 95%	P2	H1+H4+LS-CC+LS-GA	6.338	0.16%	714.0	11.90
518	1	5	8	4	0.15	10, 95%	P2	H1+H4+LS-CC+LS-GA	4.871	4.36%	745.3	12.42
519	1	5	8	2	0.00	10, 95%	P2	H1+H4+LS-CC+LS-GA	7.081	4.17%	1144.8	19.08
520	1	5	8	2	0.15	15, 95%	P2	H1+H4+LS-CC+LS-GA	6.338	0.16%	870.5	14.51
521	1	5	8	2	0.15	5, 95%	P2	H1+H4+LS-CC+LS-GA	7.105	4.99%	84.8	1.41
522	1	5	8	4	0.15	15, 95%	P2	H1+H4+LS-CC+LS-GA	4.690	0.46%	90.7	1.51
523	1	5	8	2	0.00	5, 95%	P2	H1+H4+LS-CC+LS-GA	7.789	1.25%	74.9	1.25
524	1	5	8	2	0.00	15, 95%	P2	H1+H4+LS-CC+LS-GA	7.081	4.17%	3439.9	57.33
525	1	5	8	4	0.00	5, 95%	P2	H1+H4+LS-CC+LS-GA	6.253	0.00%	126.0	2.10
526	1	5	8	4	0.15	5, 95%	P2	H1+H4+LS-CC+LS-GA	5.368	1.43%	394.3	6.57
527	1	5	8	4	0.00	10, 95%	P2	H1+H4+LS-CC+LS-GA	5.373	0.00%	363.3	6.05
528	1	5	8	4	0.00	15, 95%	P2	H1+H4+LS-CC+LS-GA	5.373	0.00%	1203.5	20.06
529	1	5	8	2	0.15	10, 95%	P3	H1+H4+LS-CC+LS-GA	5.511	0.00%	1471.2	24.52
530	1	5	8	4	0.15	10, 95%	P3	H1+H4+LS-CC+LS-GA	3.421	0.00%	322.1	5.37
531	1	5	8	2	0.00	10, 95%	P3	H1+H4+LS-CC+LS-GA	6.483	0.58%	273.0	4.55
532	1	5	8	2	0.15	15, 95%	P3	H1+H4+LS-CC+LS-GA	5.511	0.00%	878.6	14.64
533	1	5	8	2	0.15	5, 95%	P3	H1+H4+LS-CC+LS-GA	6.308	0.00%	995.6	16.59
534	1	5	8	4	0.15	15, 95%	P3	H1+H4+LS-CC+LS-GA	3.421	0.00%	365.6	6.09
535	1	5	8	2	0.00	5, 95%	P3	H1+H4+LS-CC+LS-GA	7.450	0.00%	505.0	8.42
536	1	5	8	2	0.00	15, 95%	P3	H1+H4+LS-CC+LS-GA	6.484	0.58%	324.0	5.40
537	1	5	8	4	0.00	5, 95%	P3	H1+H4+LS-CC+LS-GA	5.156	4.63%	453.5	7.56
538	1	5	8	4	0.15	5, 95%	P3	H1+H4+LS-CC+LS-GA	3.872	0.80%	249.8	4.16
539	1	5	8	4	0.00	10, 95%	P3	H1+H4+LS-CC+LS-GA	4.402	0.00%	385.1	6.42
540	1	5	8	4	0.00	15, 95%	P3	H1+H4+LS-CC+LS-GA	4.402	0.00%	353.9	5.90
541	1	5	8	2	0.15	10, 95%	P1	H1+H3+LS-CC+LS-KK	6.168	3.33%	39.3	0.65
542	1	5	8	4	0.15	10, 95%	P1	H1+H3+LS-CC+LS-KK	4.362	0.00%	40.7	0.68
543	1	5	8	2	0.00	10, 95%	P1	H1+H3+LS-CC+LS-KK	6.997	1.55%	53.1	0.88
544	1	5	8	2	0.15	15, 95%	P1	H1+H3+LS-CC+LS-KK	5.995	6.32%	36.7	0.61
545	1	5	8	2	0.15	5, 95%	P1	H1+H3+LS-CC+LS-KK	6.763	3.41%	65.7	1.09

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
546	1	5	8	4	0.15	15, 95%	P1	H1+H3+LS-CC+LS-KK	3.966	0.00%	27.7	0.46
547	1	5	8	2	0.00	5, 95%	P1	H1+H3+LS-CC+LS-KK	7.712	1.87%	83.8	1.40
548	1	5	8	2	0.00	15, 95%	P1	H1+H3+LS-CC+LS-KK	6.621	2.22%	51.6	0.86
549	1	5	8	4	0.00	5, 95%	P1	H1+H3+LS-CC+LS-KK	5.567	0.16%	58.2	0.97
550	1	5	8	4	0.15	5, 95%	P1	H1+H3+LS-CC+LS-KK	4.548	0.00%	56.3	0.94
551	1	5	8	4	0.00	10, 95%	P1	H1+H3+LS-CC+LS-KK	4.915	0.00%	40.3	0.67
552	1	5	8	4	0.00	15, 95%	P1	H1+H3+LS-CC+LS-KK	4.761	2.04%	36.6	0.61
553	1	5	8	2	0.15	10, 95%	P2	H1+H3+LS-CC+LS-KK	6.544	3.41%	25.3	0.42
554	1	5	8	4	0.15	10, 95%	P2	H1+H3+LS-CC+LS-KK	4.669	0.02%	22.8	0.38
555	1	5	8	2	0.00	10, 95%	P2	H1+H3+LS-CC+LS-KK	7.474	9.95%	35.8	0.60
556	1	5	8	2	0.15	15, 95%	P2	H1+H3+LS-CC+LS-KK	6.544	3.41%	24.1	0.40
557	1	5	8	2	0.15	5, 95%	P2	H1+H3+LS-CC+LS-KK	7.307	7.98%	36.4	0.61
558	1	5	8	4	0.15	15, 95%	P2	H1+H3+LS-CC+LS-KK	4.669	0.00%	22.3	0.37
559	1	5	8	2	0.00	5, 95%	P2	H1+H3+LS-CC+LS-KK	8.315	8.09%	46.7	0.78
560	1	5	8	2	0.00	15, 95%	P2	H1+H3+LS-CC+LS-KK	7.474	9.95%	37.4	0.62
561	1	5	8	4	0.00	5, 95%	P2	H1+H3+LS-CC+LS-KK	6.253	0.00%	43.1	0.72
562	1	5	8	4	0.15	5, 95%	P2	H1+H3+LS-CC+LS-KK	5.292	0.00%	34.0	0.57
563	1	5	8	4	0.00	10, 95%	P2	H1+H3+LS-CC+LS-KK	5.373	0.00%	27.3	0.45
564	1	5	8	4	0.00	15, 95%	P2	H1+H3+LS-CC+LS-KK	5.373	0.00%	25.7	0.43
565	1	5	8	2	0.15	10, 95%	P3	H1+H3+LS-CC+LS-KK	5.511	0.00%	39.2	0.65
566	1	5	8	4	0.15	10, 95%	P3	H1+H3+LS-CC+LS-KK	3.421	0.00%	32.3	0.54
567	1	5	8	2	0.00	10, 95%	P3	H1+H3+LS-CC+LS-KK	6.483	0.58%	48.4	0.81
568	1	5	8	2	0.15	15, 95%	P3	H1+H3+LS-CC+LS-KK	5.511	0.00%	41.7	0.70
569	1	5	8	2	0.15	5, 95%	P3	H1+H3+LS-CC+LS-KK	6.312	0.06%	55.7	0.93
570	1	5	8	4	0.15	15, 95%	P3	H1+H3+LS-CC+LS-KK	3.421	0.00%	27.5	0.46
571	1	5	8	2	0.00	5, 95%	P3	H1+H3+LS-CC+LS-KK	7.450	0.00%	71.5	1.19
572	1	5	8	2	0.00	15, 95%	P3	H1+H3+LS-CC+LS-KK	6.483	0.58%	48.1	0.80
573	1	5	8	4	0.00	5, 95%	P3	H1+H3+LS-CC+LS-KK	4.955	0.55%	58.1	0.97
574	1	5	8	4	0.15	5, 95%	P3	H1+H3+LS-CC+LS-KK	3.872	0.80%	40.2	0.67
575	1	5	8	4	0.00	10, 95%	P3	H1+H3+LS-CC+LS-KK	4.402	0.00%	44.8	0.75
576	1	5	8	4	0.00	15, 95%	P3	H1+H3+LS-CC+LS-KK	4.402	0.00%	42.5	0.71
577	1	5	8	2	0.15	10, 95%	P1	H2+H3+LS-CC+LS-RS	5.969	0.00%	53.7	0.89
578	1	5	8	4	0.15	10, 95%	P1	H2+H3+LS-CC+LS-RS	4.362	0.00%	39.9	0.67
579	1	5	8	2	0.00	10, 95%	P1	H2+H3+LS-CC+LS-RS	6.892	0.02%	63.5	1.06
580	1	5	8	2	0.15	15, 95%	P1	H2+H3+LS-CC+LS-RS	5.639	0.00%	34.9	0.58
581	1	5	8	2	0.15	5, 95%	P1	H2+H3+LS-CC+LS-RS	6.591	0.79%	57.3	0.96
582	1	5	8	4	0.15	15, 95%	P1	H2+H3+LS-CC+LS-RS	4.191	5.70%	41.7	0.69
583	1	5	8	2	0.00	5, 95%	P1	H2+H3+LS-CC+LS-RS	7.633	0.83%	90.2	1.50
584	1	5	8	2	0.00	15, 95%	P1	H2+H3+LS-CC+LS-RS	6.478	0.02%	52.7	0.88
585	1	5	8	4	0.00	5, 95%	P1	H2+H3+LS-CC+LS-RS	5.567	0.16%	67.8	1.13
586	1	5	8	4	0.15	5, 95%	P1	H2+H3+LS-CC+LS-RS	4.548	0.00%	51.8	0.86
587	1	5	8	4	0.00	10, 95%	P1	H2+H3+LS-CC+LS-RS	4.915	0.00%	45.3	0.76
588	1	5	8	4	0.00	15, 95%	P1	H2+H3+LS-CC+LS-RS	4.761	2.04%	39.6	0.66
589	1	5	8	2	0.15	10, 95%	P2	H2+H3+LS-CC+LS-RS	6.338	0.16%	31.4	0.52
590	1	5	8	4	0.15	10, 95%	P2	H2+H3+LS-CC+LS-RS	4.871	4.36%	29.3	0.49
591	1	5	8	2	0.00	10, 95%	P2	H2+H3+LS-CC+LS-RS	6.863	0.96%	26.8	0.45
592	1	5	8	2	0.15	15, 95%	P2	H2+H3+LS-CC+LS-RS	6.338	0.16%	29.5	0.49
593	1	5	8	2	0.15	5, 95%	P2	H2+H3+LS-CC+LS-RS	7.105	4.99%	64.0	1.07
594	1	5	8	4	0.15	15, 95%	P2	H2+H3+LS-CC+LS-RS	4.669	0.00%	29.9	0.50
595	1	5	8	2	0.00	5, 95%	P2	H2+H3+LS-CC+LS-RS	7.788	1.25%	37.6	0.63
596	1	5	8	2	0.00	15, 95%	P2	H2+H3+LS-CC+LS-RS	6.863	0.96%	26.9	0.45
597	1	5	8	4	0.00	5, 95%	P2	H2+H3+LS-CC+LS-RS	6.253	0.00%	54.2	0.90
598	1	5	8	4	0.15	5, 95%	P2	H2+H3+LS-CC+LS-RS	5.366	1.39%	46.2	0.77
599	1	5	8	4	0.00	10, 95%	P2	H2+H3+LS-CC+LS-RS	5.373	0.00%	30.2	0.50
600	1	5	8	4	0.00	15, 95%	P2	H2+H3+LS-CC+LS-RS	5.373	0.00%	29.1	0.49
601	1	5	8	2	0.15	10, 95%	P3	H2+H3+LS-CC+LS-RS	5.511	0.00%	44.2	0.74
602	1	5	8	4	0.15	10, 95%	P3	H2+H3+LS-CC+LS-RS	3.421	0.00%	41.8	0.70
603	1	5	8	2	0.00	10, 95%	P3	H2+H3+LS-CC+LS-RS	6.446	0.00%	63.8	1.06
604	1	5	8	2	0.15	15, 95%	P3	H2+H3+LS-CC+LS-RS	5.511	0.00%	47.4	0.79
605	1	5	8	2	0.15	5, 95%	P3	H2+H3+LS-CC+LS-RS	6.308	0.00%	63.5	1.06
606	1	5	8	4	0.15	15, 95%	P3	H2+H3+LS-CC+LS-RS	3.421	0.00%	34.8	0.58
607	1	5	8	2	0.00	5, 95%	P3	H2+H3+LS-CC+LS-RS	7.450	0.00%	134.6	2.24
608	1	5	8	2	0.00	15, 95%	P3	H2+H3+LS-CC+LS-RS	6.446	0.00%	103.8	1.73
609	1	5	8	4	0.00	5, 95%	P3	H2+H3+LS-CC+LS-RS	4.928	0.00%	81.3	1.36

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
610	1	5	8	4	0.15	5, 95%	P3	<b>H2+H3+LS-CC+LS-RS</b>	3.842	0.00%	50.3	0.84
611	1	5	8	4	0.00	10, 95%	P3	<b>H2+H3+LS-CC+LS-RS</b>	4.402	0.00%	55.2	0.92
612	1	5	8	4	0.00	15, 95%	P3	<b>H2+H3+LS-CC+LS-RS</b>	4.402	0.00%	46.8	0.78
613	1	5	8	2	0.15	10, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	5.969	0.00%	39.1	0.65
614	1	5	8	4	0.15	10, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	4.362	0.00%	33.2	0.55
615	1	5	8	2	0.00	10, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	6.890	0.00%	59.8	1.00
616	1	5	8	2	0.15	15, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	5.639	0.00%	43.2	0.72
617	1	5	8	2	0.15	5, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	6.592	0.79%	82.3	1.37
618	1	5	8	4	0.15	15, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	4.205	6.05%	30.4	0.51
619	1	5	8	2	0.00	5, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	7.673	1.36%	67.8	1.13
620	1	5	8	2	0.00	15, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	6.477	0.00%	51.8	0.86
621	1	5	8	4	0.00	5, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	5.567	0.15%	58.3	0.97
622	1	5	8	4	0.15	5, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	4.548	0.00%	74.4	1.24
623	1	5	8	4	0.00	10, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	4.915	0.00%	37.3	0.62
624	1	5	8	4	0.00	15, 95%	P1	<b>H2+H4+LS-CC+LS-RS</b>	4.761	2.04%	47.5	0.79
625	1	5	8	2	0.15	10, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	6.538	3.31%	49.1	0.82
626	1	5	8	4	0.15	10, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	5.014	7.42%	40.1	0.67
627	1	5	8	2	0.00	10, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	7.323	7.73%	55.2	0.92
628	1	5	8	2	0.15	15, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	6.522	3.06%	44.0	0.73
629	1	5	8	2	0.15	5, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	7.105	4.99%	61.5	1.02
630	1	5	8	4	0.15	15, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	4.827	3.38%	29.8	0.50
631	1	5	8	2	0.00	5, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	7.788	1.25%	39.2	0.65
632	1	5	8	2	0.00	15, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	7.323	7.73%	41.3	0.69
633	1	5	8	4	0.00	5, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	6.253	0.00%	47.3	0.79
634	1	5	8	4	0.15	5, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	5.366	1.39%	31.3	0.52
635	1	5	8	4	0.00	10, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	5.553	3.36%	29.5	0.49
636	1	5	8	4	0.00	15, 95%	P2	<b>H2+H4+LS-CC+LS-RS</b>	5.553	3.36%	34.5	0.57
637	1	5	8	2	0.15	10, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	5.511	0.00%	40.1	0.67
638	1	5	8	4	0.15	10, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	3.421	0.00%	58.3	0.97
639	1	5	8	2	0.00	10, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	6.483	0.58%	57.9	0.97
640	1	5	8	2	0.15	15, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	5.511	0.00%	59.6	0.99
641	1	5	8	2	0.15	5, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	6.308	0.00%	66.1	1.10
642	1	5	8	4	0.15	15, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	3.421	0.00%	37.0	0.62
643	1	5	8	2	0.00	5, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	7.450	0.00%	79.0	1.32
644	1	5	8	2	0.00	15, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	6.484	0.58%	68.5	1.14
645	1	5	8	4	0.00	5, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	4.955	0.55%	62.2	1.04
646	1	5	8	4	0.15	5, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	3.842	0.00%	70.2	1.17
647	1	5	8	4	0.00	10, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	4.402	0.00%	48.0	0.80
648	1	5	8	4	0.00	15, 95%	P3	<b>H2+H4+LS-CC+LS-RS</b>	4.402	0.00%	53.7	0.89
649	1	5	8	2	0.15	10, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	5.969	0.00%	143.4	2.39
650	1	5	8	4	0.15	10, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	4.362	0.00%	195.5	3.26
651	1	5	8	2	0.00	10, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	6.890	0.00%	137.6	2.29
652	1	5	8	2	0.15	15, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	5.639	0.00%	98.9	1.65
653	1	5	8	2	0.15	5, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	6.591	0.79%	125.1	2.08
654	1	5	8	4	0.15	15, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	3.966	0.00%	66.0	1.10
655	1	5	8	2	0.00	5, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	7.570	0.00%	363.9	6.06
656	1	5	8	2	0.00	15, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	6.477	0.00%	106.4	1.77
657	1	5	8	4	0.00	5, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	5.567	0.16%	226.1	3.77
658	1	5	8	4	0.15	5, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	4.548	0.00%	121.0	2.02
659	1	5	8	4	0.00	10, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	4.915	0.00%	199.4	3.32
660	1	5	8	4	0.00	15, 95%	P1	<b>H2+H3+LS-CC+LS-GA</b>	4.761	2.04%	104.0	1.73
661	1	5	8	2	0.15	10, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	6.338	0.16%	179.5	2.99
662	1	5	8	4	0.15	10, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	4.871	4.36%	112.9	1.88
663	1	5	8	2	0.00	10, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	6.863	0.96%	88.5	1.47
664	1	5	8	2	0.15	15, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	6.338	0.16%	100.9	1.68
665	1	5	8	2	0.15	5, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	7.105	4.99%	97.9	1.63
666	1	5	8	4	0.15	15, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	4.669	0.00%	134.8	2.25
667	1	5	8	2	0.00	5, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	7.788	1.25%	82.1	1.37
668	1	5	8	2	0.00	15, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	6.863	0.96%	111.1	1.85
669	1	5	8	4	0.00	5, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	6.253	0.00%	150.2	2.50
670	1	5	8	4	0.15	5, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	5.366	1.39%	127.1	2.12
671	1	5	8	4	0.00	10, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	5.373	0.00%	94.3	1.57
672	1	5	8	4	0.00	15, 95%	P2	<b>H2+H3+LS-CC+LS-GA</b>	5.373	0.00%	100.5	1.68
673	1	5	8	2	0.15	10, 95%	P3	<b>H2+H3+LS-CC+LS-GA</b>	5.511	0.00%	1713.9	28.57

Table continues on next page



Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
674	1	5	8	4	0.15	10, 95%	P3	H2+H3+LS-CC+LS-GA	3.421	0.00%	253.7	4.23
675	1	5	8	2	0.00	10, 95%	P3	H2+H3+LS-CC+LS-GA	6.446	0.00%	1812.1	30.20
676	1	5	8	2	0.15	15, 95%	P3	H2+H3+LS-CC+LS-GA	5.511	0.00%	722.0	12.03
677	1	5	8	2	0.15	5, 95%	P3	H2+H3+LS-CC+LS-GA	6.308	0.00%	1436.6	23.94
678	1	5	8	4	0.15	15, 95%	P3	H2+H3+LS-CC+LS-GA	3.421	0.00%	185.7	3.10
679	1	5	8	2	0.00	5, 95%	P3	H2+H3+LS-CC+LS-GA	7.450	0.00%	656.5	10.94
680	1	5	8	2	0.00	15, 95%	P3	H2+H3+LS-CC+LS-GA	6.446	0.00%	265.3	4.42
681	1	5	8	4	0.00	5, 95%	P3	H2+H3+LS-CC+LS-GA	4.928	0.00%	602.2	10.04
682	1	5	8	4	0.15	5, 95%	P3	H2+H3+LS-CC+LS-GA	3.842	0.00%	242.5	4.04
683	1	5	8	4	0.00	10, 95%	P3	H2+H3+LS-CC+LS-GA	4.402	0.00%	534.0	8.90
684	1	5	8	4	0.00	15, 95%	P3	H2+H3+LS-CC+LS-GA	4.402	0.00%	402.7	6.71
685	1	5	8	2	0.15	10, 95%	P1	H2+H4+LS-CC+LS-GA	5.969	0.00%	151.7	2.53
686	1	5	8	4	0.15	10, 95%	P1	H2+H4+LS-CC+LS-GA	4.362	0.00%	218.2	3.64
687	1	5	8	2	0.00	10, 95%	P1	H2+H4+LS-CC+LS-GA	6.890	0.00%	118.7	1.98
688	1	5	8	2	0.15	15, 95%	P1	H2+H4+LS-CC+LS-GA	5.639	0.00%	102.3	1.71
689	1	5	8	2	0.15	5, 95%	P1	H2+H4+LS-CC+LS-GA	6.591	0.79%	174.7	2.91
690	1	5	8	4	0.15	15, 95%	P1	H2+H4+LS-CC+LS-GA	4.191	5.70%	111.2	1.85
691	1	5	8	2	0.00	5, 95%	P1	H2+H4+LS-CC+LS-GA	7.570	0.00%	238.9	3.98
692	1	5	8	2	0.00	15, 95%	P1	H2+H4+LS-CC+LS-GA	6.477	0.00%	110.6	1.84
693	1	5	8	4	0.00	5, 95%	P1	H2+H4+LS-CC+LS-GA	5.567	0.16%	266.6	4.44
694	1	5	8	4	0.15	5, 95%	P1	H2+H4+LS-CC+LS-GA	4.548	0.00%	197.9	3.30
695	1	5	8	4	0.00	10, 95%	P1	H2+H4+LS-CC+LS-GA	4.915	0.00%	98.7	1.64
696	1	5	8	4	0.00	15, 95%	P1	H2+H4+LS-CC+LS-GA	4.761	2.04%	107.6	1.79
697	1	5	8	2	0.15	10, 95%	P2	H2+H4+LS-CC+LS-GA	6.338	0.16%	207.1	3.45
698	1	5	8	4	0.15	10, 95%	P2	H2+H4+LS-CC+LS-GA	4.871	4.36%	388.8	6.48
699	1	5	8	2	0.00	10, 95%	P2	H2+H4+LS-CC+LS-GA	7.323	7.73%	171.3	2.85
700	1	5	8	2	0.15	15, 95%	P2	H2+H4+LS-CC+LS-GA	6.338	0.16%	2014.0	33.57
701	1	5	8	2	0.15	5, 95%	P2	H2+H4+LS-CC+LS-GA	7.105	4.99%	98.8	1.65
702	1	5	8	4	0.15	15, 95%	P2	H2+H4+LS-CC+LS-GA	4.690	0.46%	2539.5	42.32
703	1	5	8	2	0.00	5, 95%	P2	H2+H4+LS-CC+LS-GA	7.788	1.25%	85.5	1.43
704	1	5	8	2	0.00	15, 95%	P2	H2+H4+LS-CC+LS-GA	7.323	7.73%	147.7	2.46
705	1	5	8	4	0.00	5, 95%	P2	H2+H4+LS-CC+LS-GA	6.253	0.00%	205.0	3.42
706	1	5	8	4	0.15	5, 95%	P2	H2+H4+LS-CC+LS-GA	5.366	1.39%	113.9	1.90
707	1	5	8	4	0.00	10, 95%	P2	H2+H4+LS-CC+LS-GA	5.373	0.00%	377.1	6.29
708	1	5	8	4	0.00	15, 95%	P2	H2+H4+LS-CC+LS-GA	5.373	0.00%	444.7	7.41
709	1	5	8	2	0.15	10, 95%	P3	H2+H4+LS-CC+LS-GA	5.511	0.00%	623.6	10.39
710	1	5	8	4	0.15	10, 95%	P3	H2+H4+LS-CC+LS-GA	3.421	0.00%	650.3	10.84
711	1	5	8	2	0.00	10, 95%	P3	H2+H4+LS-CC+LS-GA	6.446	0.00%	539.6	8.99
712	1	5	8	2	0.15	15, 95%	P3	H2+H4+LS-CC+LS-GA	5.511	0.00%	310.1	5.17
713	1	5	8	2	0.15	5, 95%	P3	H2+H4+LS-CC+LS-GA	6.308	0.00%	1950.3	32.51
714	1	5	8	4	0.15	15, 95%	P3	H2+H4+LS-CC+LS-GA	3.421	0.00%	191.1	3.18
715	1	5	8	2	0.00	5, 95%	P3	H2+H4+LS-CC+LS-GA	7.451	0.00%	785.2	13.09
716	1	5	8	2	0.00	15, 95%	P3	H2+H4+LS-CC+LS-GA	6.446	0.00%	1744.4	29.07
717	1	5	8	4	0.00	5, 95%	P3	H2+H4+LS-CC+LS-GA	4.928	0.00%	788.9	13.15
718	1	5	8	4	0.15	5, 95%	P3	H2+H4+LS-CC+LS-GA	3.842	0.00%	637.6	10.63
719	1	5	8	4	0.00	10, 95%	P3	H2+H4+LS-CC+LS-GA	4.402	0.00%	457.9	7.63
720	1	5	8	4	0.00	15, 95%	P3	H2+H4+LS-CC+LS-GA	4.402	0.00%	1189.9	19.83
721	1	5	8	2	0.15	10, 95%	P1	H2+H3+LS-CC+LS-KK	6.168	3.33%	36.5	0.61
722	1	5	8	4	0.15	10, 95%	P1	H2+H3+LS-CC+LS-KK	4.362	0.00%	38.6	0.64
723	1	5	8	2	0.00	10, 95%	P1	H2+H3+LS-CC+LS-KK	6.997	1.55%	46.7	0.78
724	1	5	8	2	0.15	15, 95%	P1	H2+H3+LS-CC+LS-KK	5.995	6.32%	33.9	0.56
725	1	5	8	2	0.15	5, 95%	P1	H2+H3+LS-CC+LS-KK	6.763	3.41%	61.5	1.03
726	1	5	8	4	0.15	15, 95%	P1	H2+H3+LS-CC+LS-KK	3.966	0.00%	26.5	0.44
727	1	5	8	2	0.00	5, 95%	P1	H2+H3+LS-CC+LS-KK	7.712	1.87%	90.0	1.50
728	1	5	8	2	0.00	15, 95%	P1	H2+H3+LS-CC+LS-KK	6.621	2.22%	41.4	0.69
729	1	5	8	4	0.00	5, 95%	P1	H2+H3+LS-CC+LS-KK	5.567	0.16%	62.6	1.04
730	1	5	8	4	0.15	5, 95%	P1	H2+H3+LS-CC+LS-KK	4.548	0.00%	51.8	0.86
731	1	5	8	4	0.00	10, 95%	P1	H2+H3+LS-CC+LS-KK	4.915	0.00%	43.3	0.72
732	1	5	8	4	0.00	15, 95%	P1	H2+H3+LS-CC+LS-KK	4.761	2.04%	36.8	0.61
733	1	5	8	2	0.15	10, 95%	P2	H2+H3+LS-CC+LS-KK	6.719	6.18%	26.7	0.44
734	1	5	8	4	0.15	10, 95%	P2	H2+H3+LS-CC+LS-KK	4.669	0.02%	19.9	0.33
735	1	5	8	2	0.00	10, 95%	P2	H2+H3+LS-CC+LS-KK	7.495	10.27%	32.1	0.53
736	1	5	8	2	0.15	15, 95%	P2	H2+H3+LS-CC+LS-KK	6.719	6.18%	25.7	0.43
737	1	5	8	2	0.15	5, 95%	P2	H2+H3+LS-CC+LS-KK	7.307	7.98%	34.6	0.58

Table continues on next page

Test	$R$	$D$	$T$	$g$	$q$	$y, \text{Snorm}(y)$	$\chi, \lambda$	Heuristic	Total deferred patients	Deviation from CE+CE	Runtime (seconds)	Runtime (minutes)
738	1	5	8	4	0.15	15, 95%	P2	<b>H2+H3+LS-CC+LS-KK</b>	4.669	0.00%	23.4	0.39
739	1	5	8	2	0.00	5, 95%	P2	<b>H2+H3+LS-CC+LS-KK</b>	8.315	8.09%	48.1	0.80
740	1	5	8	2	0.00	15, 95%	P2	<b>H2+H3+LS-CC+LS-KK</b>	7.495	10.27%	33.9	0.56
741	1	5	8	4	0.00	5, 95%	P2	<b>H2+H3+LS-CC+LS-KK</b>	6.253	0.00%	50.5	0.84
742	1	5	8	4	0.15	5, 95%	P2	<b>H2+H3+LS-CC+LS-KK</b>	5.366	1.39%	39.2	0.65
743	1	5	8	4	0.00	10, 95%	P2	<b>H2+H3+LS-CC+LS-KK</b>	5.373	0.00%	25.9	0.43
744	1	5	8	4	0.00	15, 95%	P2	<b>H2+H3+LS-CC+LS-KK</b>	5.373	0.00%	25.3	0.42
745	1	5	8	2	0.15	10, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	5.511	0.00%	38.3	0.64
746	1	5	8	4	0.15	10, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	3.421	0.00%	27.7	0.46
747	1	5	8	2	0.00	10, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	6.446	0.00%	48.3	0.81
748	1	5	8	2	0.15	15, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	5.511	0.00%	40.3	0.67
749	1	5	8	2	0.15	5, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	6.312	0.06%	56.0	0.93
750	1	5	8	4	0.15	15, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	3.421	0.00%	26.5	0.44
751	1	5	8	2	0.00	5, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	7.450	0.00%	74.9	1.25
752	1	5	8	2	0.00	15, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	6.446	0.00%	47.7	0.79
753	1	5	8	4	0.00	5, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	4.928	0.00%	54.0	0.90
754	1	5	8	4	0.15	5, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	3.842	0.00%	41.3	0.69
755	1	5	8	4	0.00	10, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	4.402	0.00%	44.2	0.74
756	1	5	8	4	0.00	15, 95%	P3	<b>H2+H3+LS-CC+LS-KK</b>	4.402	0.00%	42.5	0.71

# Appendix E

## Numerical results in the case study

In this appendix we present the final results per iteration of our algorithm (combination of Heuristics 1, 3, LS-CC and LS-RS).

Iteration $n$	Day $d$	Appointment requests $\lambda^d$	Deferred patients		Capacity cycle $k^d$	CAS $C^d$
			Iteration $n - 1$	Iteration $n$		
1	1	12.023	0	0.237	14	(2,1,1,2,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,2,0,0,0,0,0,1,0,0,1,0)
	2	11.944	0	0.188	10	(2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,1,0,0,0,0,0,0,0,0,0,0)
	3	11.626	0	0.157	10	(2,1,0,2,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0)
	4	13.460	0	0.225	10	(2,1,1,2,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0)
	5	10.282	0	0.155	16	(1,1,2,1,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,2,1,0,1,0,0,1,0,1,1,0,0)
2	1	12.260	0.237	0.221	14	(2,1,1,2,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,2,0,0,0,0,0,1,1,0,0,0)
	2	12.132	0.188	0.184	10	(2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0)
	3	11.783	0.157	0.151	10	(2,2,0,1,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0)
	4	13.685	0.225	0.218	10	(2,1,1,2,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0)
	5	10.437	0.155	0.137	17	(2,2,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,0,1,1,0,0,1,1,1,0,0)
3	1	12.244	0.221	0.221	14	(2,1,1,2,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,2,0,0,0,0,0,1,1,0,0,0)
	2	12.127	0.184	0.184	10	(2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0)
	3	11.777	0.151	0.151	10	(2,2,0,1,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0)
	4	13.677	0.218	0.209	10	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0)
	5	10.419	0.137	0.137	17	(2,2,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,0,1,1,0,0,1,1,1,0,0)
4	1	12.244	0.221	0.219	14	(2,1,1,2,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,2,0,0,0,0,0,1,0,1,0,0)
	2	12.127	0.184	0.184	10	(2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0)
	3	11.777	0.151	0.148	10	(2,2,1,1,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0)
	4	13.669	0.209	0.209	10	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0)
	5	10.419	0.137	0.135	17	(2,2,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,1,0,1,0,0,1,1,1,0,0)
5	1	12.242	0.219	0.206	14	(2,2,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,2,0,0,0,0,0,1,0,1,0,0)
	2	12.127	0.184	0.184	10	(2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0)
	3	11.774	0.148	0.148	10	(2,2,1,1,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0)
	4	13.669	0.209	0.209	10	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0)
	5	10.418	0.135	0.132	17	(2,2,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0)
6	1	12.229	0.206	0.201	14	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,2,0,0,0,0,0,1,1,0,0,0)
	2	12.127	0.184	0.184	10	(2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0)
	3	11.774	0.148	0.148	10	(2,2,1,1,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0)
	4	13.669	0.209	0.209	10	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0)
	5	10.414	0.132	0.132	17	(2,2,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0)
7	1	12.224	0.201	0.200	14	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,2,0,0,0,0,0,1,0,1,0,0)
	2	12.127	0.184	0.184	10	(2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0)
	3	11.774	0.148	0.147	10	(2,2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0)
	4	13.669	0.209	0.209	10	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0)
	5	10.414	0.132	0.132	17	(2,2,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0)
8	1	12.223	0.200	0.199	14	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,2,1,0,0,0,0,1,0,1,0,0)
	2	12.127	0.184	0.184	10	(2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0)
	3	11.773	0.147	0.147	10	(2,2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0)
	4	13.669	0.209	0.209	10	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0)
	5	10.414	0.132	0.132	17	(2,2,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0)
9	1	12.222	0.199	0.199	14	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,2,1,0,0,0,0,1,0,1,0,0)
	2	12.127	0.184	0.184	10	(2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0)
	3	11.773	0.147	0.147	10	(2,2,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0)
	4	13.669	0.209	0.209	10	(2,2,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0)
	5	10.414	0.132	0.132	17	(2,2,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0)

## Appendix F

# Case study test under 85% workload

In this appendix we show the results that we obtained in a first test where the workload is 85%. We present a table with performance measures and show graphs with the appointment schedules plotted against the walk-in arrival rate.

Capacity cycle method	Day schedule method	Fraction of walk-in patients served on day of arrival	Runtime (minutes)	Fraction of walk-in patients served on day of arrival	Runtime (minutes)
		Workload 62.3%		Workload 85.0%	
<b>H1+LS-CC</b>	<b>H3+LS-RS</b>	99.43%	82.9	94.16%	192.2
<b>H1+LS-CC</b>	<b>Benchmark 1</b>	97.58%	5.1	86.24%	14.2
<b>H1+LS-CC</b>	<b>Benchmark 2</b>	97.58%	5.1	86.24%	14.2
<b>H2+LS-CC</b>	<b>Benchmark 3</b>	97.67%	5.0	86.78%	12.0

Table F.1: Performance of the algorithm and benchmarks with a workload of 85%

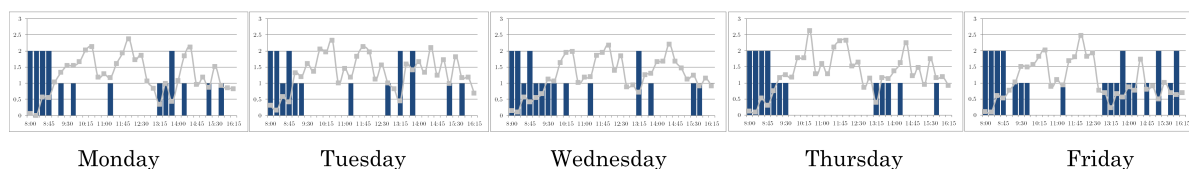


Figure F.1: Appointment schedule and walk-in arrival rate per day with a workload of 85%