Flowing through hospitals

Johanna Theresia van Essen

Flowing through hospitals

Johanna Theresia van Essen

Dissertation committee

Chairman & secretary:	Prof. dr. ir. A.J. Mouthaan
Promotor:	Prof. dr. J.L. Hurink
Members:	Prof. dr. K.I. Aardal
	Prof. dr. R.J. Boucherie
	Prof. dr. ir. E.W. Hans
	Dr. M. van Houdenhoven
	Prof. dr. S. Nickel

Ph.D. dissertation, University of Twente, Enschede, the Netherlands
Center for Telematics and Information Technology (No. 13-272, ISSN 1381-3617)
Beta Research School for Operations Management and Logistics (No. D171)
Center for Healthcare Operations Improvement and Research

This research was financially supported by the Dutch Technology Foundation STW by means of the project 'Logistical Design for Optimal Care' (No. 08140)

Printed by Ipskamp Drukkers, Enschede, the Netherlands

ISBN 978-90-365-0725-7
DOI 10.3990/1.9789036507257

FLOWING THROUGH HOSPITALS

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
Prof. dr. H. Brinksma,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op donderdag 21 november 2013 om 14.45 uur

door

Johanna Theresia van Essen

geboren op 27 juli 1985
te Dordrecht, Nederland

Dit proefschrift is goedgekeurd door de promotor:

Prof. dr. J.L. Hurink

Voor pappa

Dankwoord

De afgelopen vier jaren waren een spannende, maar vooral leuke tijd. De afwisseling tussen werken in het HagaZiekenhuis in Den Haag, de Universiteit Twente, Duitsland en nog vele andere locaties maakte dat ik mij geen moment heb verveeld. Maar het voornaamste wat deze tijd zo leuk maakte, waren alle mensen om mij heen die ik graag wil bedanken voor hun rol in het tot stand komen van dit proefschrift.

Allereerst natuurlijk mijn promotor Johann zonder wie ik nooit aan dit traject begonnen was. Het vak Scheduling wat ik bij jou volgde, was één van de leukste vakken tijdens mijn master. Toen je bij het mondeling aangaf dat jij mij een geschikte kandidaat voor promoveren vond, begon ik hierover na te denken. En toen aan het eind van mijn master deze leuke promotieplek beschikbaar kwam, was de keuze snel gemaakt. Tijdens mijn promotietijd kon ik altijd bij je binnenlopen om advies te vragen. Je (vaak onleesbare) commentaar op mijn artikelen heeft mijn schrijfstijl een stuk verbeterd. Bedankt voor de fijne samenwerking de afgelopen jaren en alle gezellige informele gesprekken.

Ook de overige commissieleden wil ik graag bedanken. Richard en Erwin, het was fijn om tijdens mijn promotie deel uit te maken van CHOIR en met jullie samen te werken. Mark, bedankt dat jij mij wegwijs hebt gemaakt in het HagaZiekenhuis en bedankt voor je stimulerende enthousiasme voor alle projecten waaraan ik heb meegewerkt in het HagaZiekenhuis. Karen, leuk dat jij als mijn afstudeerprofessor nu ook deel uitmaakt van mijn promotiecommissie. Stefan, thank you for the opportunity to visit Karlsruhe and for taking part in my committee.

Mijn collega's in het HagaZiekenhuis wil ik ook graag bedanken. Toen ik in het begin nog geen vaste werkplek had, werd ik hartelijk welkom geheten door mijn collega's bij Beleid & Kwaliteit. Bedankt voor jullie interesse in mij en mijn onderzoek. Auke, Arnoud en Lisanne, het was leuk om samen met jullie het Logistieke Bedrijf van het HagaZiekenhuis op te starten en samen te werken aan verschillende projecten. Ook de vele afstudeerders wil ik bedanken voor de samenwerking, afleiding en bijdrage aan dit proefschrift.

Mijn collega's op de UT wil ik bedanken voor de gezelligheid, adviezen en nuttige discussies de afgelopen jaren. Ik had het geluk om mij bij drie groepen thuis te voelen: CHOIR, DWMP en SOR. Aleida, het was op de UT altijd een stuk gezelliger als jij er was. Bedankt voor het grondig doornemen van dit proefschrift op zoek naar fouten en inconsistenties. Egbert, we zijn ongeveer tegelijk begonnen aan ons promotietraject en daarom vind ik het erg leuk om dit samen af te kunnen sluiten. Het was altijd erg gezellig op de UT en congressen. Harm, bedankt voor de fijne samenwerking bij LNMB vakken en de werkcolleges van Discrete Wiskunde.

Maartje van de Vrugt, leuk dat jij de laatste paar jaar deel was van CHOIR. Bedankt dat er altijd een bed voor me klaarstond tijdens het laatste jaar van mijn promotie. Maartje Zonderland, jij was als mijn grote zus op de UT. Bedankt voor je bezoek in Karlsruhe en je bijdrage aan de titel van dit proefschrift. Nardo, onze samenwerking was kort maar leuk. Fijn dat jij mijn werk in het HagaZiekenhuis voortzet.

Also thanks to my colleagues in Karlsruhe. You made me feel really welcome by organizing game nights and diners. It is nice to still be in touch with some of you.

Mijn huisgenoten aan de Benjamin Willem Ter Kuilestraat hebben de vele donderdagavonden in Enschede een stuk leuker gemaakt. Esther, Carolien, Annemiek en Lisette, bedankt voor alle gezelligheid en tv-avonden.

De afgelopen vier jaar hebben mijn vrienden voor de nodige afleiding gezorgd. Hilde, Hester en Ariette, bedankt voor alle gezellige avonden en uitjes. En alle studiegenoten, leuk dat we elkaar nog regelmatig zien.

En tot slot wil ik mijn familie bedanken. Lieve pappa, ook al hebt u mijn promotietraject niet kunnen meemaken, u hebt mij van jongs af aan aangemoedigd en gestimuleerd. Zonder u had ik dit nooit kunnen bereiken. Lieve familie, als je uit zo'n groot gezin komt en er ook nog een grote schoonfamilie bij komt, besteed je veel weekenden aan verjaardagen en familiefeestjes. Ik geniet hier ontzettend van en het was altijd een welkome onderbreking van werk. Wijnie, bedankt voor het oppassen het afgelopen half jaar en dat er altijd een bordje eten voor me klaarstond. Lieve zus, bedankt dat ik regelmatig mijn rit op vrijdag van Enschede naar Den Haag kon onderbreken voor een warme maaltijd en gezelligheid bij jullie in Amersfoort. Herbert en Rick, bedankt dat jullie mijn paranimfen willen zijn. Het voelt goed om twee van mijn grote broers achter me te hebben staan. En natuurlijk ook mamma en Laurus, bedankt voor jullie steun en liefde.

Lieve Helena, jij was tijdens het laatste jaar van mijn promotie in mijn leven. Eerst in mijn buik in Duitsland en later tijdens het afronden van dit proefschrift in levende lijve. Jouw aanwezigheid heeft geholpen om dingen in perspectief te blijven zien en het leven weer zoveel uitdagender gemaakt. Lieve Daan, dat wij samen ons promotietraject hebben doorgemaakt was een bijzondere periode. Het was fijn om alle ups en downs met jou te kunnen delen en bedankt dat jij mij altijd aanmoedigt mijn ambities te verwezenlijken.

Theresia van Essen
Den Haag, oktober 2013

Contents

Part I

Introduction

Chapter 1

Research motivation and outline

1.1 Challenges in healthcare

Due to an aging population and increased healthcare costs, hospitals are forced to use their resources more efficiently, meaning that the same number of patients has to be treated with less resources or more patients with the same amount of resources. The mentioned resources are, for example, the Operating Room (OR), the beds at the wards, and ambulances. Practitioners in the hospitals often think that the quality of care will reduce when resources are used more efficiently, because they believe that efficiency means that, for example, patients are discharged too soon and surgeries are rushed. However, efficiency means that, for example, the utilization of resources is increased or the length of stay (LOS) and surgery duration are reduced (if possible), but not at the cost of quality of care. On the contrary, the reduction achieved by improving the efficiency may also improve the quality of care as, e.g., unnecessary waiting time is reduced or even eliminated.

Improving the efficiency in hospitals is challenging as many uncertainties have to be taken into account. It already starts with the arrival of patients, as it cannot be predicted when a patient gets ill and needs medical attention. Furthermore, when a patient enters the hospital it is not known yet what the diagnosis is and what clinical path the patient will follow. This holds in principle for the entire stay of a patient as it is never certain what the next step in the clinical path will be. And even if the steps would be completely known, the LOS remains uncertain as one patient may need more time to recover than another patient. All these uncertainties together make planning in healthcare more challenging than in most industries.

1.2 Flowing through hospitals

The efficiency of healthcare can be improved by carefully scheduling and planning the processes at the departments the patient visits during his stay. Therefore, procedures or appointments should be scheduled such that, e.g., (1) the waiting time for the patients is minimized, (2) the schedule is robust against changes in the procedure or appointment time, and (3) the arrival of emergency patients is accounted for. The waiting time for patients should be minimized to limit the chance on complications, and thereby, increase the chance on full recovery. However, reducing the waiting time for one patient group might increase the waiting time of another patient group. Therefore, changing things in the schedule should be done carefully

to make sure that the waiting time for each patient group stays somehow proportional to the severity of the patients' condition. In addition, reducing the waiting time for one department may not always lead to a reduction of the entire LOS of a patient. For example, if the waiting time for an outpatient clinic visit is reduced, but not the waiting time until surgery, this reduction has no effect on the patients' state of health. Thus, minimizing waiting time has to be done with care.

The extreme results of long waiting times may be the cancellation of an already scheduled procedure when, e.g., the process within an OR or outpatient clinic gets disrupted too much because of a longer procedure or appointment time, it may happen that one or more patients get canceled. In addition, when a patient's LOS is longer than expected, it may happen that another scheduled patient cannot be admitted or has to wait longer for surgery since, e.g., the intensive care may be fully occupied. This increases the waiting time for this patient and has a negative effect on the patients' state of mind, and thus, the chance on complications increases. A possible solution to these problems is to create a robust schedule which can deal with these disruptions in a good way, thereby reducing the chance on canceling a patient.

Another type of disruption is caused by emergencies. When the arrival of emergency patients has not been accounted for, these patients have to wait some time before they can be treated or an elective patient has to be canceled such that an emergency patient can take his place. Obviously, both situations are not preferred, and therefore, the occurrence of such an event should be prevented. To make sure that an emergency patient can be treated immediately, or within a short amount of time, some capacity in the OR and other departments may be reserved. However, by blocking time for emergency patients, the waiting time for elective patient increases while the reserved time might not be fully used by the emergency patients. Therefore, accounting for emergency patients when creating schedules is a delicate task.

In this dissertation, we mainly focus on solution methods that have as goal to minimize waiting times, create robust schedules, and account for the arrival of emergency patients. For some problems considered in this dissertation, one or two of these points are more important than the other points. This can be given by the nature of the problem or can be imposed by the hospital at which the problem originated. Sometimes, the objectives of a problem are even conflicting, i.e, reducing the waiting time might decrease the robustness of the schedule. When one or more (conflicting) objectives are considered, a balanced trade-off has to be made. One possible way to deal with this conflict is to provide a weight for each objective such that the objectives are considered in a correct way.

1.3 Scheduling in healthcare

Most of the considered problems in this dissertation are related to scheduling. Scheduling is a decision-making process that deals with the allocation of resources to tasks and its goal is to optimize one or more objectives [78]. The resources can

be, e.g., machines, crews, or operating rooms and the tasks can be, e.g., production operations, services, or surgeries. Since several decades scheduling is also applied in the healthcare sector, for example in surgery [18] and ambulance scheduling [65]. For an overview of all types of applications in healthcare see [52].

Scheduling is usually done on the operational level. However, this dissertation also discusses problems on the strategic and tactical level. The strategic level is concerned with end-to-end optimization of patient flows and dimensioning individual departments within a time horizon of one or more years. The tactical level is concerned with allocating available resources to groups of patients that share the same characteristics from a medical and logistics point of view in a time-horizon of a few weeks up to a few months. The operational level is concerned with, e.g., planning and scheduling a given demand of elective patients within a time-horizon of a few days up to a few weeks while taking into account several uncertainties such as arriving emergency patients and stochastic treatment durations. On the operational level often a distinction is made between off-line and on-line. On the operational off-line level procedures are planned in advance and the operational on-line level deals with monitoring the process and reacting to unforeseen or unanticipated events [46].

On each of these three levels, uncertainty plays an important role, and therefore, robust schedules have to be developed. On the strategic level the demand for the coming year is very uncertain although good assumptions can be made based on data of previous years. However, the development of new equipment and treatments may change the demand for care. On the tactical level, the way in which the demand is distributed over the days or months is still uncertain even though the total demand might be known. In addition, the resource availability is uncertain, and therefore, schedules which allow slack should be developed. At the operational level, we have to deal with uncertain procedure times and the arrival of emergency patients. This means that schedules must be able to deal with changes during the day.

As the problems in healthcare become very complex due to uncertainties, exact solution methods are often not able to solve the models within an acceptable amount of time. Therefore, heuristic solution methods play an important role, also in this dissertation. Heuristics aim to find a good solution to the problem within a reasonable amount of time. However, they do not guarantee that an optimal solution will be found.

In this dissertation, we discuss various solution methods that deal with the uncertainties characteristic for healthcare. According to the considered planning level, different uncertainties have to be taken into account. We show how to deal with these uncertainties and how to create robust solutions.

1.4 Applied research environment

The research discussed in this dissertation is conducted at various hospitals in the Netherlands, namely HagaZiekenhuis, Isala Clinics, and Erasmus Medical Cen-

ter. These three hospitals are comparable when considering the number of admissions and number of outpatient clinic visits in 2012 as shown in Table 1.1. The HagaZiekenhuis and Erasmus Medical Center are both one of the 11 trauma centers in the Netherlands and serve an urban area. The Erasmus Medical Center is the largest academic hospital in the Netherlands and all three hospitals provide top clinical care.

Even though the problems considered in this dissertation originated at a particular hospital, all developed solutions can also be implemented in any hospital as the models are set up generically.

	HagaZiekenhuis	Isala Clinics	Erasmus Medical Center
# Employees	4128	11824	5714
# Beds	660	1320	944
# Clinical admissions	38105	41773	47428
# Day care admissions	34465	42043	51140
# Outpatient visits	580346	519907	557867
Average LOS (days)	5.3	6.9	4.8

Table 1.1: Statistics of the considered hospitals over the year 2012

1.5 Dissertation outline

This dissertation consists of four parts. Part I consists of this chapter and Chapter 2 which informally introduces combinatorial optimization problems and how these problems can be characterized with respect to their complexity. In addition, the basic solution methods used in this dissertation are introduced and explained.

In Figure 1.1, the structure of the remaining chapters is depicted. Part II consists of Chapter 3 and discusses ambulance planning. Chapter 4 and Chapter 5 build up Part III and discuss problems concerning operating room planning. Part IV, consisting of Chapter 6 and 7, completes this dissertation and discusses planning problems concerning the beds at the wards.

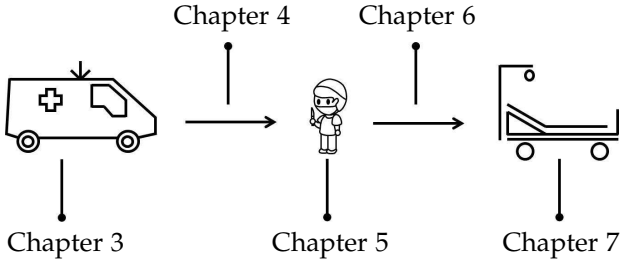


Figure 1.1: Outline dissertation

In Part II of this dissertation we consider one of the ways in which patients arrive at the hospital, namely by ambulance. Patients who arrive by ambulance need immediate care, and therefore, it is important that they arrive at the hospital as soon as possible. This implies that ambulances must be located such that a given response time can be met. Not only the location of the ambulances is important to guarantee this response time, but also the number of ambulances available at each ambulance base. In Chapter 3, we discuss solution approaches that determine the locations of ambulance bases and the number of ambulances needed at each base.

Patients who arrive by ambulance may need surgery. As these patients cannot be scheduled beforehand, they have to break in into the elective surgery schedule. When there is no operating room fully reserved for emergency patients, the patient has to wait until one of the operating rooms becomes available. Methods that minimize the waiting time for emergency surgeries are discussed in Chapter 4.

When such an emergency surgery is started, the elective surgery schedule is disrupted. The OR-schedule can also be disrupted when surgeries take longer than expected. In Chapter 5, we discuss a decision support system which suggests changes in the OR-schedule such that the disruption and number of cancellations are minimized.

After surgery, most patients are admitted at one of the wards to recover. To guarantee that a bed is available after surgery, the bed availability must be taken into account when the OR-schedule is created. In this way, the bed occupancy can be leveled such that the number of beds needed is minimized. In Chapter 6, solution methods for this problem are discussed.

Another important factor concerning the bed availability is the way in which the clinical departments are distributed over the wards. When more than one clinical department share a ward, also the risk of refusing a patient is shared, and thereby, reduced. However, not all clinical departments can share a ward due to medical reasons. In Chapter 7, several solution approaches are introduced which assign clinical departments to wards such that enough beds are available and such that all medical constraints are fulfilled.

Chapter 2

Combinatorial optimization problems and solution methods

In this chapter, we informally introduce general solution methods used in this dissertation to solve the considered problems. First, some background information about the type of problems considered is given, and afterwards, the solution methods to solve these problems are discussed. To explain the solution methods, an example problem is used. The aim of this example problem is to demonstrate the solution methods, but not to find the best solution method for it.

2.1 Combinatorial optimization problems

The problems discussed in this dissertation are all combinatorial optimization problems. These are problems for which a finite set of solutions exists together with an objective value for each solution and for which we want to find the best solution in this finite set of solutions. This best solution is referred to as the optimal solution. One classical example of a combinatorial optimization problem is the knapsack problem which is defined as follows. Let us consider a knapsack which can carry a maximum weight W. In addition, a set of items I is given and each of these items has weight w_i and profit p_i. For simplicity, we assume that all weights and profits are non-negative, i.e., $w_i \geq 0$ and $p_i \geq 0$ for all $i \in I$. The problem is to determine a subset $\bar{I} \subseteq I$ of these items that maximizes the total profit $\sum_{i \in \bar{I}} p_i$ such that the total weight of the items is less than or equal to the maximum weight the knapsack can carry, i.e., $\sum_{i \in \bar{I}} w_i \leq W$.

We consider a small example of the knapsack problem depicted in Figure 2.1 to further explain the definition of a combinatorial optimization problem. Let an instance of the knapsack problem be given where the knapsack has capacity 13 and can be filled with four items: an apple, a compass, a sandwich, and a bottle of water. The first item, the apple, has weight five and profit six, i.e., $w_1 = 5$ and $p_1 = 6$. The compass has weight two and profit eight, i.e., $w_2 = 2$ and $p_2 = 8$, the sandwich has weight two and profit two, i.e., $w_3 = 2$ and $p_3 = 2$, and the bottle of water has weight eight and profit ten, i.e., $w_4 = 8$ and $p_4 = 10$. It can easily be seen that there are only a finite number of solutions to this problem. First note that we can either decide to choose zero, one, two, three, or four items. When we add all four items to the knapsack, the total weight $\sum_{i \in I} w_i$ equals 17 which is more than the knapsack can carry. Therefore, this solution is called infeasible.

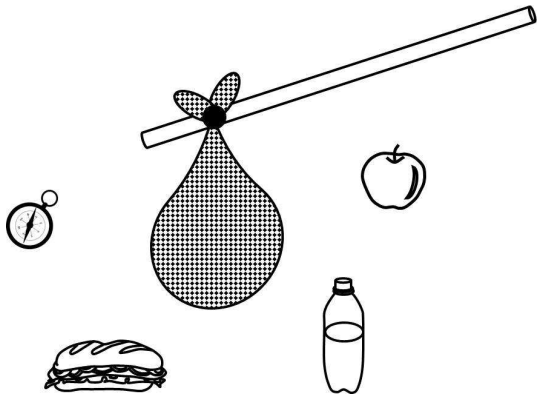


Figure 2.1: Knapsack problem

In Table 2.1, all possible solutions are enumerated. It shows that 13 of the 16 possible solutions are feasible, i.e., the total weight of these solutions is less than or equal to the capacity of the knapsack. From this finite set of feasible solutions, we select one with the highest profit as the optimal solution. For this example, this is the solution which adds the compass, sandwich, and water to the knapsack.

Solution	Total weight	Total profit	Feasibility
Empty knapsack	0	0	Feasible
Apple	5	6	Feasible
Compass	2	8	Feasible
Sandwich	2	2	Feasible
Water	8	10	Feasible
Apple, compass	7	14	Feasible
Apple, sandwich	7	8	Feasible
Apple, water	13	16	Feasible
Compass, sandwich	4	10	Feasible
Compass, water	10	18	Feasible
Sandwich, water	10	12	Feasible
Apple, compass, sandwich	9	16	Feasible
Apple, compass, water	15	24	Infeasible
Apple, sandwich, water	15	18	Infeasible
Compass, sandwich, water	12	20	Feasible
Apple, compass, sandwich, water	17	26	Infeasible

Table 2.1: Set of solutions to example

With only three items it is easy to determine an optimal solution to the knapsack problem as the number of solutions only equals eight. When we have n items,

the total number of solutions equals $\sum_{k=0}^{n} \binom{n}{k} = 2^n$. This means that when we have ten items, the total number of solutions equals $2^{10} = 1024$ and for 30 items, we already have $2^{30} = 1,073,741,824$ solutions. Thus, the number of solutions increases exponentially in n, and therefore, also the solution time if we would enumerate all solutions.

2.2 Complexity

The exponential growth of the number of solutions given the instance size n indicates that the knapsack problem might be hard to solve when the problem instances get large. However, specific large instances might still be easy to solve. To determine whether a given combinatorial optimization problem is easy or hard, complexity classes are introduced. To explain the notion of complexity classes, we first have to explain the distinction between optimization problems and decision problems. Recall that an optimization problem is defined as the problem of finding an optimal solution to the problem amongst a finite set of feasible solutions. A decision problem does not aim to find an optimal solution but answers a question which can only be answered by 'yes' or 'no'. Each optimization problem can easily be transformed into a decision problem by introducing a bound P on the value of an optimal solution. For the knapsack problem, the corresponding decision problem is: does there exist a solution with total profit greater than or equal to P such that the total weight is less than or equal to W?

The complexity classes $\mathcal{P}$ and $\mathcal{NP}$ refer to the complexity of decision problems. The class $\mathcal{P}$ denotes the class of decision problems that can be solved in polynomial time (in the input size). A decision problem is in the class $\mathcal{NP}$ (non-deterministic polynomial time) if an answer to a 'yes'-instance of the decision problem can be verified in polynomial time. The decision problem of the knapsack problem is clearly in $\mathcal{NP}$ as it can easily be verified whether a given solution indeed leads to a 'yes'-answer or not.

Because the decision problems in $\mathcal{P}$ can be solved in polynomial time, a 'yes'-answer can be verified in polynomial time. Therefore, the definitions of classes $\mathcal{P}$ and $\mathcal{NP}$ imply that $\mathcal{P}$ is a subset of $\mathcal{NP}$, i.e, $\mathcal{P} \subseteq \mathcal{NP}$. The question whether $\mathcal{P} = \mathcal{NP}$ or $\mathcal{P} \neq \mathcal{NP}$, i.e., can all solutions that can be verified in polynomial time also be found in polynomial time or not, is one of the seven millennium prize problems. Solving this problem will be rewarded with a million dollar prize.

$\mathcal{NP}$-complete problems play an important role in solving this open millennium prize problem. These problems form a class of decision problems that are equivalent in the sense that either all or none of these problems can be solved by a polynomial time algorithm. A problem is said to be $\mathcal{NP}$-complete if all problems in the class $\mathcal{NP}$ can be transformed to this problem in polynomial time. This makes the $\mathcal{NP}$-complete problems the hardest problems in $\mathcal{NP}$. If there is an $\mathcal{NP}$-complete problem which can be solved within polynomial time then $\mathcal{P} = \mathcal{NP}$. If $\mathcal{P} \neq \mathcal{NP}$ then no $\mathcal{NP}$-complete problem can be solved in polynomial time.

To prove whether a given decision problem in $\mathcal{NP}$ is $\mathcal{NP}$-complete it suffices

to transform an already known $\mathcal{NP}$-complete problem to this problem in polynomial time. Therefore, we only need a first $\mathcal{NP}$-complete problem which then can be transformed to the considered problem. This first $\mathcal{NP}$-complete problem was found by Cook [22], namely the satisfiability problem.

To prove that the decision version of the knapsack problem is $\mathcal{NP}$-complete, we have to make a polynomial time transformation from a well-known $\mathcal{NP}$-complete problem to the knapsack problem. The problem we consider is the subset-sum problem [36] which is defined as follows. Given n positive integers $a_1, \ldots, a_n$, does there exist a subset $\bar{I} \subset \{1, \ldots, n\}$ such that $\sum_{i \in \bar{I}} a_i = K$? Recall that the decision problem of the knapsack problem is defined as follows. Given n items with positive weights w_i and positive profits p_i, does there exist a subset $\bar{I} \subset \{1, \ldots, n\}$ such that $\sum_{i \in \bar{I}} p_i \geq P$ and $\sum_{i \in \bar{I}} w_i \leq W$. Thus, we can easily transform an instance of the subset-sum problem to the knapsack problem by setting $w_i = p_i = a_i$ and $W = P = K$. Then, the question for the knapsack problem becomes: does there exist a subset $\bar{I} \subset \{1 \ldots, n\}$ such that $\sum_{i \in \bar{I}} a_i \geq K$ and $\sum_{i \in \bar{I}} a_i \leq K$? Both summations together lead to $\sum_{i \in \bar{I}} a_i = K$ which is equal to the subset-sum problem. This transformation is surely polynomial, and therefore, proves that the knapsack problem is $\mathcal{NP}$-complete.

The complexity classification of decision problems is transferred to optimization problems by calling an optimization problem $\mathcal{NP}$-hard, if its corresponding decision problem is $\mathcal{NP}$-complete. When an optimization problem is $\mathcal{NP}$-hard it is not likely to find an efficient exact algorithm. However, the size or properties of an instance might be such that exact solution methods are fast enough to solve the problem within a reasonable amount of time. For all other cases, we may use heuristic solution methods to solve the problem. Heuristic solution methods do not guarantee to find an optimal solution, but aim to find a good solution within a reasonable amount of time. These two types of solution methods are further discussed in the next sections.

2.3 Exact solution methods

An exact solution method guarantees that an optimal solution to a problem is found. One possible exact solution method is to enumerate all possible solutions and select one of these as the optimal solution like we did in Section 2.1. However, there exist methods that do not enumerate all possible solutions, but instead discards large subsets of solutions which do not contain an optimal solution. Such methods take less time than full enumeration. One of these faster exact solution methods is branch-and-bound [76].

To apply branch-and-bound to the knapsack problem, we first formulate it as an Integer Linear Program (ILP). To formulate the knapsack problem as an ILP, we have to introduce binary variables that indicate whether an item is chosen to be in the knapsack or not. For this purpose, we introduce binary variables X_i which are one when item $i \in I$ is included and zero otherwise. Using these variables, the restriction for the weight given by $\sum_{i \in \bar{I}} w_i \leq W$, can be expressed by $\sum_{i \in I} w_i X_i \leq$

W. The objective function to maximize the total profit, i.e., $\max \sum_{i \in \bar{I}} p_i$, is then given by $\max \sum_{i \in I} p_i X_i$. The complete ILP is given by:

$$\begin{aligned} \max \quad & \sum_{i \in I} p_i X_i \\ \text{s. t.} \quad & \sum_{i \in I} w_i X_i \leq W, \\ & X_i \in \{0,1\}, \qquad \forall i \in I. \end{aligned} \tag{2.1}$$

Note that the binary variables X_i specify a solution to the problem.

The branch-and-bound technique divides the set of feasible solutions into smaller and smaller subsets. This is called the branching step. The next step is the bounding step that determines a bound on the objective function value of such subsets of feasible solutions. If this bound indicates that this subset does not contain an optimal solution, then this subset is discarded. When the considered problem contains binary variables it is easy to branch on the two possible values of one of the binary variables. One branch fixes the value of the considered binary variable to zero and the other branch fixed this value to one. This branching leads to a solution tree which specifies the considered subsets of feasible solutions. The starting point for building up the tree is the set of all feasible solutions. A possible bound on the objective function values within this set can be achieved by solving the Linear Program (LP) relaxation of the considered problem. The LP-relaxation of an ILP is given by allowing the integer variables in (2.1) to take continuous values. Binary variables are relaxed by allowing them to take any value between zero and one. Thus, the LP-relaxation of the knapsack problem is:

$$\begin{aligned} \max \quad & \sum_{i \in I} p_i X_i \\ \text{s. t.} \quad & \sum_{i \in I} w_i X_i \leq W, \\ & 0 \leq X_i \leq 1, \qquad \forall i \in I. \end{aligned} \tag{2.2}$$

In general, an LP can be solved in polynomial time by, for example, the simplex method that was introduced by Dantzig [23]. As our main focus is solving ILPs we do not further explain this method. However, the LP-relaxation of the knapsack problem can be solved by an algorithm that is easier than the simplex method. For each item considered, we compute the profit of the item per unit weight, i.e., $\frac{p_i}{w_i}$. Next, we sort the items in non-increasing order according to these values. Adding the item with the highest profit per unit weight results in the highest total profit. Therefore, we add the items in this order until the maximum weight is achieved. The last item might not be fully added to the knapsack, and thus, for the LP-relaxation it might happen that only a fraction of this item is added to the knapsack. Let us consider the same example as discussed before and for which the $\frac{p_i}{w_i}$-values are depicted in Table 2.2.

Item	Weight	Profit	$\frac{p_i}{w_i}$
1. Apple	5	6	$\frac{6}{5}$
2. Compass	2	8	4
3. Sandwich	2	2	1
4. Water	8	10	$\frac{5}{4}$

Table 2.2: Example

When we add the items to the knapsack in non-increasing order of $\frac{p_i}{w_i}$, we add the compass and bottle of water completely, i.e., $X_2 = X_4 = 1$, and $\frac{3}{5}$ part of the apple, i.e., $X_1 = \frac{3}{5}$. The total weight of the added items then equals $2 + 8 + \frac{3}{5} \cdot 5 = 13$ and the total profit equals $8 + 10 + \frac{3}{5} \cdot 6 = 21\frac{3}{5}$. As this is an optimal solution for the LP-relaxation, the objective function value provides an upper bound on the optimal solution to the original ILP. We also have a lower bound that is given by the solution which only adds the compass and bottle of water to the knapsack as this is a feasible integer solution. The total profit for this solution is 18.

The solution to the LP-relaxation is the starting point of our tree. The next step is to branch. We choose to branch on the item with the fractional value, in our case the first item which is the apple. For the first branch, X_1 is set to zero and for the second branch, X_1 is set to one. For both subsets of solutions, we solve the LP-relaxation. The value for X_1 is fixed, but the other values can be freely chosen within the bounds stated by (2.2). The steps of the branch-and-bound method for our instance of the knapsack problem are described below. Note that we only have evaluated five solutions instead of the total $2^4 = 16$ solutions. The entire branch-and-bound tree for this example is depicted in Figure 2.2.

Step 1. Solve the LP-relaxation. The solution to the LP-relaxation is $\boldsymbol{X} = \left(\frac{3}{5}, 1, 0, 1\right)$ and the resulting upper bound is $8 + 10 + \frac{3}{5} \cdot 6 = 21\frac{3}{5}$. The lower bound for the total profit of the optimal solution is $8 + 10 = 18$.

Step 2. Branch on X_1. The left branch considers the situation with X_1 set to zero and the right branch considers the situation with X_1 set to one.

Step 3. Solve the LP-relaxation for both branches.

- Step (a) The solution to the LP-relaxation for the left branch is $\boldsymbol{X} = (0, 1, 1, 1)$ with upper bound and lower bound equal to $8 + 2 + 10 = 20$. Note that this now is the new lower bound on the optimal objective function value. Because the solution to the LP-relaxation is also feasible for the ILP, and thus, the lower bound and upper bound are equal to each other, we do not further explore this branch.
- Step (b) The solution to the LP-relaxation for the left branch is $\boldsymbol{X} = \left(1, 1, 0, \frac{3}{4}\right)$ with upper bound equal to $6 + 8 + \frac{3}{4} \cdot 10 = 21\frac{1}{2}$ and

lower bound equal to $6 + 8 = 14$. As the upper bound is higher than the lower bound 20, we might still be able to find a better integer solution than the solution found in the left branch. Therefore, we further branch on the variable X_4.

Step 4. Solve the LP-relaxation with X_1 set to one and X_4 set to zero or one.

Step (a) The solution to the LP-relaxation for the left branch is $\boldsymbol{X} = (1, 1, 1, 0)$ with upper bound and lower bound equal to $6 + 8 + 2 = 16$. As the upper bound and lower bound are both lower than the bound provided by the integer solution $\boldsymbol{X} = (0, 1, 1, 1)$, we do not further explore this branch.

Step (b) The solution to the LP-relaxation for the right branch is $\boldsymbol{X} = (1, 0, 0, 1)$ with upper bound and lower bound equal to $6 + 10 = 16$. As the upper bound and lower bound are both lower than the bound provided by the integer solution $\boldsymbol{X} = (0, 1, 1, 1)$, we do not further explore this branch.

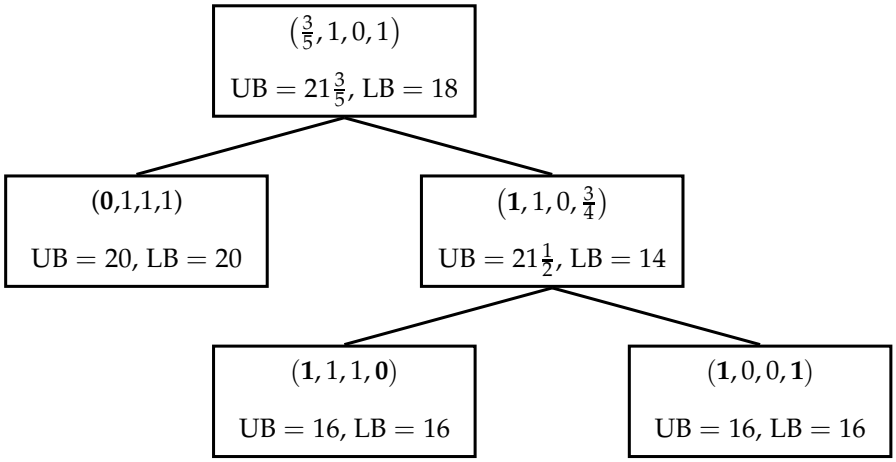


Figure 2.2: Branch-and-bound tree for knapsack example

Note that there exist more exact solution methods to solve combinatorial optimization problems, for example Dynamic Programming [76]. However, we do not further discuss these methods as they are not used in this dissertation.

2.4 Heuristic solution methods

As the exact solution method described in the previous section might be too time consuming for large instances of $\mathcal{NP}$-hard problems, we may consider heuristic solution methods to solve these problems. These methods aim to find a good

solution to the problem within a reasonable amount of time. Often it holds that allowing more computation time for an heuristic solution method results in a better solution to the original problem. Therefore, a trade-off has to be made between the computation time and the quality of the solution. In this section, we discuss several heuristic solution methods which are used in this dissertation.

2.4.1 Constructive heuristics

Constructive heuristics are heuristics which generate single solutions to the problem. These type of heuristics start with an empty solution and build up this solution until a complete solution is obtained. The way in which the empty solution is build up to a complete solution depends on the problem considered. In Section 2.3, we already introduced such a constructive heuristic to determine the lower bound to the ILP problem. This heuristic sorts the items in non-increasing order of the $\frac{p_i}{w_i}$-values and adds the items to the knapsack until the next item does not fit anymore.

2.4.2 Local search heuristics

As the solutions obtained by constructive heuristics often are not optimal or even not good, local search heuristics can be used to improve these solutions. A local search heuristic starts with an initial solution and attempts to find a better solution in the neighborhood of this initial solution. This procedure is then repeated in an iterative manner until some stopping criterion is fulfilled. The neighborhood of a solution is defined as a set of solutions that can be obtained through an allowed modification.

For the knapsack problem, the neighborhood of a solution is, for example, defined as all solutions for which one item is added or for which one item is added and one item is removed. When we consider the example introduced in Section 2.3 the initial solution is (0,1,0,1) with a total profit of 18. Thus, the neighborhood of this solution and its corresponding values is as given in Table 2.3.

Solution	Total weight	Total profit	Feasibility
(1,1,0,1)	15	24	Infeasible
(0,1,1,1)	12	20	Feasible
(1,0,0,1)	13	16	Feasible
(0,0,1,1)	10	12	Feasible
(1,1,0,0)	7	14	Feasible
(0,1,1,0)	4	10	Feasible

Table 2.3: Neighborhood of initial solution (0,1,0,1)

Table 2.3 shows that the best solution in the neighborhood of (0,1,0,1) is (0,1,1,1) with a total profit of 20 which is more than the total profit of (0,1,0,1). Therefore, we

accept (0,1,1,1) as the new current solution. The neighborhood of (0,1,1,1) consists of three infeasible solutions and one feasible solution with a total profit of 16. Therefore, the local search procedure stops at this point.

During a local search procedure it can happen (and normally happens) that we get stuck in a local optimum instead of a global optimum. In the following, we introduce two local search heuristics which attempt to leave local optima and come closer to the global optimum.

Simulated annealing

The name of and inspiration for Simulated Annealing (SA) (see [1], [32], [92]) originate from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce defects. The heat causes the atoms to move from their initial positions to other random states of higher energy. The slow cooling gives them more chance of finding configurations with lower internal energy than the initial one.

By analogy with this physical process, each step of SA possibly moves from the current solution to a randomly selected neighbor solution. If the neighbor solution has a better objective function value than the current solution, the neighbor solution is accepted. Otherwise, the neighbor solution is accepted with probability $e^{\frac{-\Delta}{T}}$, where Δ is the difference between the objective value of the current and the neighbor solution, and T is the current temperature. If the neighbor solution is rejected, SA stays at the current solution. Note that the probability of accepting a worse solution is almost proportional to the difference in objective values since $e^{\frac{-\Delta}{T}} \sim 1 - \frac{\Delta}{T}$. Slightly worse solutions have a reasonably high probability of being accepted, while much worse solutions are only accepted infrequently. The temperature T gradually decreases during the search process, and therefore, also the acceptance probability of a worse solution decreases. The allowance of moving to worse solutions makes it possible to escape from a (poor) local optimum.

To apply SA, the stopping criterion and cooling scheme must be specified. The cooling scheme needs to specify the initial value of T and how this temperature is updated in each step of the procedure. The initial temperature T_0 is normally chosen such that in the first iterations most of the neighbor solutions have a relatively high probability of being accepted. The updating of the temperature T is often done by decreasing it in every iteration by a factor α with $0 < \alpha < 1$. For each temperature level, we perform several iterations which form a Markov chain, because the next state only depends on the current state. The length of the Markov chain, indicated by L, should be chosen such that it is proportional to the size of the neighborhood (see e.g. [1]). As stopping criterion, we may set a threshold T_f for the temperature. This threshold T_f is chosen such that worse solutions have a low probability of being accepted. SA can be summarized by the following algorithm, where $\bar{S}$ denotes the current best solution.

Step 1. Generate an initial solution S using some constructive heuristic and determine its objective function value $f(S)$. Set $\bar{S} = S$. Set an initial temperature

T_0 and a reduction factor α. Set $T = T_0$.

Step 2. Repeat L times:

Step (a) Select a neighbor solution S' of solution S at random and determine $f(S')$.

Step (b) If $f(S')$ is better than $f(S)$, set $S = S'$, and if $f(S')$ is better than $f(\bar{S})$, set $\bar{S} = S'$.
If $f(S')$ is worse than $f(S)$, set $S = S'$ with probability $e^{\frac{-\Delta}{T}}$.

Step 3. Set $T = \alpha T$. If $T < T_f$, then stop. Else, go to Step 2.

Tabu search

Like SA, Tabu Search (TS) (see [32], [40]) moves from one solution to a neighbor solution with the new solution being possibly worse than the one before. The main difference between TS and SA is the mechanism used for accepting a candidate solution. In TS, the mechanism is not probabilistic but deterministic, because it systematically searches the neighborhood and selects the best solution found, even if this solution is worse than the current solution. During the process, a tabu list is kept which contains solutions, or properties of the solutions, the heuristic is not allowed to accept. In the most simple case, the tabu list has a fixed number of entries. Every time a new solution is accepted, the current solution, or some property of the current solution, enters the tabu list and the oldest entry is deleted. The tabu list aims to avoid returning to a solution that has been visited previously. TS can be summarized by the following algorithm, in which $\bar{S}$ is the current best solution.

Step 1. Generate an initial solution S using some constructive heuristic and determine its objective function value, $f(S)$. Set $\bar{S} = S$.

Step 2. Select one of the best neighbor solutions S' of solution S that is not tabu. Solution S', or a property of solution S', enters the tabu list and the oldest entry is deleted. Set $S = S'$.

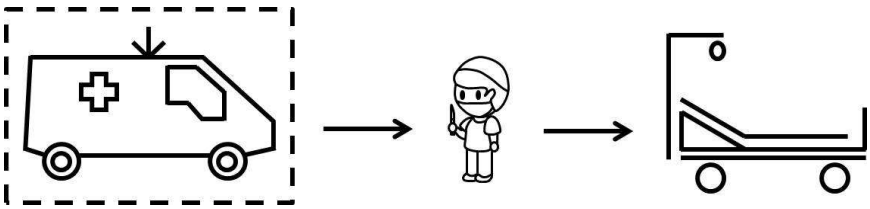
Step 3. If $f(S')$ is better than $f(\bar{S})$, then set $\bar{S} = S'$.

Step 4. If the stopping criterion is met, then stop. Else, go to Step 2.

Note that there exist more heuristic solution methods to solve combinatorial optimization problems, for example Genetic Algorithms [71]. However, we do not further discuss these methods as they are not used in this dissertation.

Part II

Ambulance planning



Chapter 3

Models for ambulance planning on the strategic and the tactical level

3.1 Introduction

Emergency Medical Service (EMS) systems as they exist in the U.S., Canada, or European countries like the Netherlands are very complex. When planning such a system, there are lots of different aspects that have to be considered and many questions have to be answered, for example legal regulations, regional distinctions, or geographical characteristics. Usually, the problem of planning the EMS system can be divided into smaller subproblems as location planning, dispatching and so forth which are generally easier to solve than the overall problem. However, the subproblems depend on one another such that the solution of one subproblem forms the basis for solving the next one. Combining all the partial solutions then defines the planning of the EMS system. One of the subproblems within EMS systems is the question where to locate bases and ambulances throughout the considered region. This can either be done with a prefixed number of available ambulances or the location decisions can be made simultaneously while determining the number of needed vehicles. The problem of locating ambulances and ambulance bases can be divided into three phases: the strategic, the tactical, and operational level. At the strategic level the locations of the ambulance bases are determined while considering constraints on the coverage. In the next step, the tactical level, the explicit number of ambulances needed per base to fulfill all demand is specified. At the operational level, the allocation of ambulances to emergencies and relocation of ambulances must be carried out in real-time. In this chapter, we focus on the first two levels as the operational level differs significantly from the other two and should be covered separately.

In this work, we first present a solution approach for solving the strategic and tactical planning problem simultaneously. This approach is mainly used as a benchmark to be able to evaluate solutions and computation times, as we suggest to split the problem into the two levels mentioned above. The reason for this splitting is that solving the strategic and tactical level simultaneously leads to a complex problem with long computing times. However, as the location of (larger) bases is often fixed for years but the number and location of ambulances can change each year, it seems to be a logic decision to solve the two levels separately. Nevertheless, when a replanning of an EMS system is wanted and the location of bases together

with number of ambulances should be determined simultaneously, the proposed simultaneous approach can be applied to tackle the problem.

The chosen solution approach for solving both levels at once is a stochastic programming formulation. The input instances for this formulation are quite large, and therefore, the problem has to be simplified more than is the case when we solve both levels separately. However, solving the problem in two stages may result in a suboptimal solution. Therefore, we compare the solutions of the stochastic programming formulation with the solutions of the approach that solves the problem in two stages.

At the strategic level, we determine the locations of the ambulance bases. When locating these ambulance bases, we also have to take into account the location of the emergency departments as not only the driving time between the patient location and the ambulance location should be minimized, but also the total driving time from an ambulance location to the hospital location via the patient location might be of interest. Therefore, when the patient location is far from the hospital location, it might be necessary to locate an ambulance base close to the patient to minimize this total driving time.

The number of ambulances needed at each base is a decision at the tactical level as this may change regularly according to changes in demand. The decision at the tactical level highly depends on the time-dependent demand and travel time, and therefore, we propose to use simulation to solve this problem. We start with an initial number of ambulances based on average demand and travel time. This number is adapted iteratively by simulating the current situation and suggesting moves to improve the current solution. In other words, we incorporate simulation in a local search approach. In the simulation, we also consider the covering constraints as mentioned in the model for the strategic level.

In contrast to many of the probabilistic approaches presented in literature like the maximum expected covering location problem introduced by Daskin [25], the maximum availability location problem introduced by ReVelle and Hogan [81] or the stochastic formulation by Beraldi et al. [10], we chose scenarios to model the uncertainty instead of defining busy fractions for the ambulances or making the calls occurring randomly. In addition, the models presented in this chapter have a different way of modeling the coverage constraints. Based on the idea presented by Gendreau et al. [37] that simple coverage might not be sufficient, coverage constraints are modeled generic such that different levels of coverage can be included.

Concluding, the contribution of this chapter is as follows: we decompose the ambulance planning problem into a strategic and tactical level, present a formal description of the problem at each level, and introduce and compare methods for solving both levels separately and in an integrated way.

The chapter is structured as follows: Section 3.2 covers a literature review. In Section 3.3, we present the problems at the two stages, give corresponding formulations, and discuss possible shortcomings. The proposed solution approaches are presented in Section 3.4. First, the overall approach and second, the approaches for the two separated levels are given. The results of comparing the models are

discussed in Section 3.5. The chapter closes with a summary and an outlook in Section 3.6.

3.2 Literature review

In ambulance location planning, there already exists a large variety of literature. We do not aim to review all developed approaches, and therefore, we only discuss the approaches most relevant for our research. For a complete overview of related literature, the reader is referred to surveys as they can be found in Marianov and ReVelle [68], Owen and Daskin [75], Brotcorne and Laporte [14], Galvao et al. [35] and Li et al. [65].

Concerning the modeling of the coverage constraints, several formulations are discussed in literature which can be used for the problem on the strategic level. The first emergency base location covering model in literature is the location set covering model (LSCM) which was introduced by Toregas et al. [89]. Its objective is to find the minimum number of ambulance bases needed to cover all demand points. Several other covering models such as the maximal cover location problem (MCLP) introduced by Church and ReVelle [21], the double standard model (DSM) introduced by Gendreau et al. [37], the maximum expected covering location problem (MEXCLP) introduced by Daskin [25], and the maximum availability location problem (MALP) introduced by ReVelle and Hogan [81] all assume a fixed number of available bases, and thereby, can only indirectly be used to minimize the number of bases. In addition, the DSM and MEXCLP models already assign several vehicles to each base to guarantee the coverage, i.e., solve the strategic and tactical problem at the same time. An interesting covering model is the gradual coverage model introduced by Karasakal and Karasakal [55]. This model uses a sigmoid function to model the gradual decline of coverage along with an increase of the distance, and thereby, relaxes the 'all or nothing' assumption when a fixed radius is specified. Berman et al. [12] proposed the cooperative coverage model which assumes that a demand point is covered when the total received 'signal' from several bases exceeds a certain threshold. This means that when a certain demand point lies further away from a base, a second base may be needed to fulfill this threshold.

All the models mentioned above are deterministic and static. The first probabilistic approach was presented by Chapman and White [19] in 1974. It was a probabilistic set covering model in which servers were not always available. Aly and White [5] published a formulation for the probabilistic set covering problem together with a variation of it in 1978. They assumed the location of incidents to be random variables. As mentioned above, Daskin [25] proposed in 1983 the MEXCLP. There, he included the idea that an ambulance is busy for a fraction of time. He assumed that the number of ambulances that have been placed on the network was given. As an extension of the MALP, Marianov and ReVelle [69] developed the Queuing MALP or Q-MALP in 1996. They used results from queuing theory to relax the assumption that the busyness probabilities of different servers are independent. In addition, travel times were also considered to be random variables.

Among other probabilistic approaches that for example use reliability constraints and busy fractions for servers as done by ReVelle and Hogan [80], there are two main approaches for including stochasticity into the ambulance location problem, namely hypercube queuing models and stochastic programming. The first hypercube queuing model was introduced by Larson [61] in 1974. Based on that, different variations can be found for example in [39], [53], [54], [83], or [87]. Beraldi et al. [10] present a stochastic integer problem formulation under probabilistic constraints (SIPC) that determines where service sites must be located and how many emergency vehicles must be assigned to each site while randomness in the demand of emergency services is assumed. They give a deterministic equivalent formulation of the introduced constraints using the so-called p-efficient points of a joint probability distribution function. Beraldi and Bruni [9] propose a stochastic programming model under probabilistic constraints as a two-stage approach. They relax the assumption of server independence and assume randomness in the emergency requests instead of in the server availability. Noyan [73] developed two types of stochastic optimization models involving alternate risk measures, the first one including integrated chance constraints (ICC) and the second one incorporating ICCs and a stochastic dominance constraint. He modeled the random demands using the scenario approach and relaxed the assumptions that the service providers operate independently and that the demand sites are independent of each other. The stochastic program presented in Section 3.4 of this chapter is based on the formulations by Beraldi and Bruni [9] and Noyan [73]. In contrast to Beraldi and Bruni [9] we do not enforce that each demand location has to be served by only one ambulance base. In addition, we extend the general two-stage formulation Noyan described [73] by generic coverage constraints.

There is also quite some literature considering simulation for ambulance planning because determining the number of needed ambulances highly depends on the time-dependent demand and travel time. Several of the simulation studies, e.g., [34], [41], [47], and [50] use simulation to evaluate location policies determined by optimization models. Swoveland et al. [86] use simulation in combination with a form of branch-and-bound to determine the location of ambulances. Berlin and Liebman [11] use simulation to evaluate location policies, but also to determine the number of ambulances needed. The assigned number of ambulances per base is set sufficiently high to serve all requests and after the simulation it is determined how many ambulances are needed to guarantee availability in, for example, 95% of the time. This idea is incorporated in the simulation approach developed in this chapter to determine the number of needed ambulances at the tactical level. Zaki et al. [98] use simulation to determine the effect of using an ambulance from a different region on the average response time and overall coverage. The simulation shows that the average response time increases, but also the overall coverage increases.

To the best of our knowledge, existing literature lacks an explicit integration of the different levels for ambulance planning in EMS systems together with a comparison of separated and combined solution approaches (for the strategic and the tactical level). In addition, existing literature tends to present formulations

(and algorithms) specified only for certain EMS systems. General and generic formulations and approaches are needed to enable a comparison between different systems, for example. This work is supposed to start filling these gaps.

3.3 Problem formulation

In this section, formulations for the planning problems at the strategic and the tactical level are presented. We first discuss the problem of locating ambulance bases, and after that, the problem of determining the number of ambulances per base is introduced.

3.3.1 Strategic level

When an accident happens, an ambulance is sent to the location of the accident to provide first aid and to transport the patient to the hospital. To limit the risk of medical complications, the patient should arrive at the hospital as soon as possible. Because the locations of hospitals are fixed, the time until a patient arrives at the hospital can only be influenced by the time it takes for an ambulance to arrive at the patient's location as the treatment time needed at the scene cannot be influenced. In addition, the sooner an ambulance arrives at the accident location, the sooner the medical treatment can start.

In general, ambulances are stationed at ambulance bases. An ambulance drives from its base location to the patient's location and after transporting the patient to the hospital, the ambulance returns to its base. Most countries have some coverage requirements for the locations of the ambulance bases. For example, time limits can be given for the maximum time an ambulance is allowed to need to arrive at the patient's location or for the time period till a patient arrives at the hospital for further treatment.

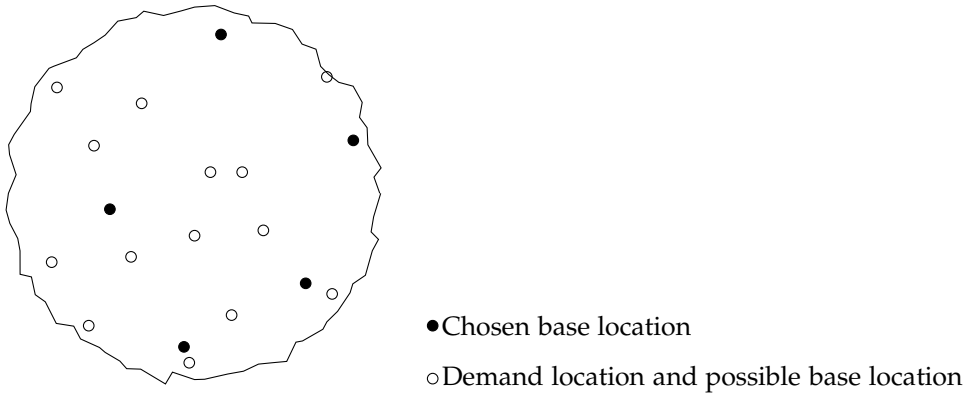


Figure 3.1: The locations of bases are determined on the strategic level

At the strategic level, the aim is to minimize the number of ambulance base locations while fulfilling the given coverage requirements as show in Figure 3.1. To formalize this problem, we use the following notation of which an overview can be found in the appendix (Section 3.7). The set of demand locations, i.e., the set of locations where an accident might happen, is given by set I. The demand for each demand location $i \in I$ is given by d_i and can either be stochastic or deterministic. For now, we assume that the demand is fixed and given. The set of potential base locations is given by set J and from these potential base locations, we have to select a subset such that all coverage requirements are fulfilled. We model this by introducing a binary variable X_j which is one when ambulance base location $j \in J$ is selected and zero otherwise.

To specify the coverage requirements, for each demand location $i \in I$, we specify a subset of base locations that lie within the range of the considered coverage requirement. As the coverage mostly only depends on the driving time between the demand and ambulance base location, these subsets can easily be determined beforehand. When, for example, the driving time from the base location via a demand location to the hospital location is limited by a coverage requirement, the coverage for a certain demand location only depends on the driving time between the demand and base location. Because the driving time from the demand location to the hospital location is fixed, this can easily be subtracted from the total driving time. Often, more than only one coverage requirement is considered. For this, we introduce a set K of coverage requirements. For example, there might be different maximal allowed driving times for varying severity of the incidents, resulting in several different coverage requirements, or double coverage requirements as proposed by Gendreau et al. [37] might be given. The subset of base locations which fulfill coverage requirement $k \in K$ for demand location $i \in I$ is denoted by $J_{ki} \subseteq J$. To determine whether demand location $i \in I$ is covered according to coverage requirement $k \in K$, we introduce binary variables Y_{ki} that take value one if for demand location $i \in I$ the coverage constraint $k \in K$ is fulfilled and zero otherwise. The following constraint ensures that these binary variables Y_{ki} take value zero if the coverage requirement is not fulfilled:

$$\sum_{j \in J_{ki}} X_j \geq Y_{ki}, \qquad \forall i \in I, k \in K. \tag{3.1}$$

In general, for most coverage constraints $k \in K$, not 100% of the demand but only a (large) fraction of the demand has to be covered. In addition, this fraction could be specified for all locations $i \in I$ together (the entire country) or specific subsets of locations (regions). The latter ensures that all regions in a country have the same coverage, while in the first situation, one region could be covered less than another region. Furthermore, different coverage requirements may focus on different levels. For example, a first coverage requirement should hold for the entire country, a second coverage requirement for each state in a country, and a third coverage requirement for each municipality in a country. Therefore, we

introduce for each coverage requirement $k \in K$ a set of regions R_k for which the coverage requirement must hold. More precisely, for each coverage requirement $k \in K$, we partition the set of demand locations I into $|R_k|$ subsets denoted by I_{kr} with $r \in R_k$, i.e., for each region $r \in R_k$ we specify the demand locations $i \in I$ that lie within this region. As the fraction of demand covered according to coverage requirement $k \in K$ does not have to be the same for each region $r \in R$, we introduce α_{kr} as the fraction of demand to be covered in region $r \in R$ according to coverage requirement $k \in K$. This fraction can, for example, be smaller for a region in which certain demand locations are hard to reach because they lie on a mountain top or on an island. In addition, in case that coverage requirements differ for urban and rural areas (e.g., by law), this also can be modeled by the introduced constraints. More formally, the following constraint ensures that a fraction α_{kr} of the demand in region $r \in R_k$ is covered according to coverage requirement $k \in K$:

$$\sum_{i \in I_{kr}} d_i Y_{ki} \geq \alpha_{kr} \sum_{i \in I_{kr}} d_i, \quad \forall k \in K, r \in R_k. \tag{3.2}$$

Note that constraint (3.2) may become irrelevant for some coverage requirements $k \in K$ as it may be dominated by one of the other coverage requirements. To illustrate this, let us consider two coverage requirements $k_1, k_2 \in K$. When $J_{k_1 i} \subseteq J_{k_2 i}$ for all $i \in I$ and $\alpha_{k_1 r} \geq \alpha_{k_2 r}$, then coverage requirement k_2 is dominated by coverage requirement k_1. However, for most practical instances this situation will not occur for all demand locations $i \in I$, but only for a subset of the demand locations, because the base locations included in J_{ki} may for example depend on varying driving times per demand location.

As the aim at the strategic level is to minimize the number of chosen ambulance base locations, our objective function can be formulated as follows:

$$\min \sum_{j \in J} X_j. \tag{3.3}$$

Summarizing, the problem for locating bases at the strategic level looks as follows:

$$\begin{aligned}
\min \quad & \sum_{j \in J} X_j && && (3.4)\\
\text{s. t.} \quad & \sum_{j \in J_{ki}} X_j \geq Y_{ki}, && \forall i \in I, k \in K,\\
& \sum_{i \in I_{kr}} d_i Y_{ki} \geq \alpha_{kr} \sum_{i \in I_{kr}} d_i, && \forall k \in K, r \in R_k,\\
& X_j, Y_{ki} \in \{0,1\}, && \forall i \in I, j \in J, k \in K.
\end{aligned}$$

It is easy to see that in practice there are some shortcomings of the model. Note that if demand location $i \in I$ is covered by one of the selected ambulance base locations, this does not necessarily mean that this demand location is always covered

in practice, because the coverage also depends on the ambulance availability at this base location. In addition, it may happen that the nearest hospital to a demand location has insufficient capacity, and thus, the patient has to be transported to another hospital. In other words, the strategic level does not consider the varying ambulance and hospital capacity. Therefore, we have to extend our model to also include these aspects within the tactical level.

3.3.2 Tactical level

In constraint (3.2) it is ensured that the full demand d_i of a location $i \in I$ is covered if at least one base location $j \in J$ with $j \in J_{ki}$ is opened. Thus, it is somehow assumed that always enough ambulances are available at the opened base locations. However, as this is in general not the case, we determine at the tactical level the number of ambulances needed to fulfill the considered coverage requirements as shown in Figure 3.2. Note that also the hospital capacity can be a restriction, but as this capacity depends on much more than only the emergency calls, we do not include this at this level.

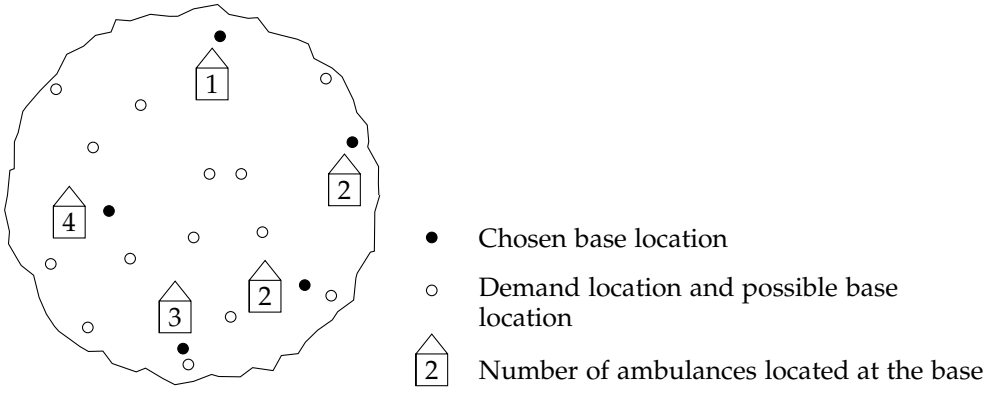


Figure 3.2: The numbers of ambulances per base location are determined on the tactical level

Before we introduce the model to determine the number of needed ambulances, we first clarify the situation at the tactical level by providing the following sketch of the handling of an emergency call. When an emergency call occurs, we first might have to wait until one of the ambulances becomes available. When an ambulance is or becomes available, it drives from the ambulance base location to the location of the emergency call. Note that for the tactical level we assume that ambulances always start at their bases when driving to an emergency and that they always go back there afterwards. Of course, this assumption is only valid for the tactical level and would not be applied on the operational level where the actual location of the ambulance has to be taken into account. The driving time from the base

location to the call location depends on the traffic intensity at that moment and has influence on the coverage. An example for a coverage requirement could be that an ambulance should arrive at the emergency location within 15 minutes after the emergency call occurred. This means that the call is covered when the waiting time plus the stochastic driving time from base to call location is less than or equal to 15 minutes. When the ambulance arrives at the emergency call location, the patient is first treated at the scene and, if necessary, placed in the ambulance. This time between the arrival and departure of the ambulance at the emergency call location is called the treatment time and depends on the injuries of the patient. After the treatment time, the patient is transported to the hospital if necessary. This driving time also depends on the traffic intensity at that moment. Another coverage requirement could be that a patient should arrive at the hospital within 45 minutes after the call occurred. This means that an emergency call is covered according to this coverage requirement when the waiting time plus the driving time from base to call location plus the treatment time and the driving time from call to hospital location is less than or equal to 45 minutes. Note that this time includes much uncertainty. When the ambulance arrives at the hospital, the patient is transferred to the hospital personnel. This also takes some time which differs per patient and injury. After the transfer, the ambulance can return to its base location and becomes available, possibly after a cleaning process, for the next emergency call. Figure 3.3 visualizes this example.

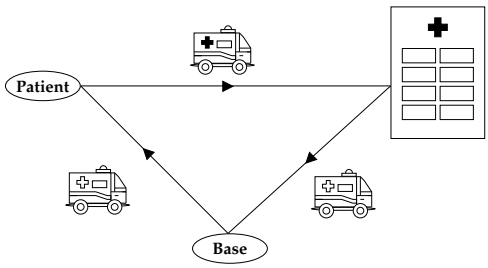


Figure 3.3: A typical workflow on the tactical level

To model the sketched situation at the tactical level, we first have to generate incoming emergency calls. The arrivals of emergency calls can be modeled in several ways, but for now we assume that the arriving emergency calls are known beforehand. We denote this set of emergency calls by E and each emergency call $e \in E$ has an arrival time given by t_e and a location denoted by l_e. Recall that the location l_e of emergency call $e \in E$ is an element of the set of demand locations I. Then, each emergency call should be served by one of the base locations chosen at the strategic level. Note that for the strategic level, set J represented the set of potential ambulance base locations, while for the tactical level, set J denotes the set of selected ambulance base locations. To assign emergency call $e \in E$ to one of the available base locations $j \in J$, we introduce binary variables Z_{ej} which are one when emergency call $e \in E$ is served by ambulance base $j \in J$. The following constraints

ensure that each emergency call $e \in E$ is assigned to exactly one ambulance base location:

$$\sum_{j \in J} Z_{ej} = 1, \qquad \forall e \in E. \tag{3.5}$$

An ambulance base location can only be assigned to an emergency call when there is an ambulance available to serve it. For this, it sometimes might be necessary to let a patient wait until an ambulance becomes available. This waiting time for emergency call $e \in E$ is denoted by a variable W_e. This waiting time W_e has to be chosen such that an ambulance becomes available at base location $j \in J$ after this waiting time. The ambulance then has to drive to the emergency call location, pick up the patient, and transport the patient to the hospital. Finally, the ambulance returns to its base location. As this total driving time depends on the varying traffic intensity, the varying treatment time at the emergency location, and the varying transfer time at the hospital, we represent this driving time by a stochastic parameter v_{ej} which denotes the time an ambulance from base $j \in J$ is occupied when it is assigned to emergency call $e \in E$. For now, we assume that this value can be determined beforehand for each pair of emergency call $e \in E$ and ambulance base location $j \in J$. When all emergency calls are assigned to an ambulance base location and the waiting times for emergency calls $e \in E$ are determined, we can calculate how many ambulances from base location $j \in J$ are occupied at each moment in time. To formalize this, we discretize the considered planning horizon in a set of time points denoted by T. The number of ambulances at base location $j \in J$ occupied at time $t \in T$ is now represented by an integer variable A_{jt} for which the correct value is determined by the following constraint:

$$A_{jt} = \sum_{e \in E} Z_{ej} \mathbb{1}_{\{t_e + W_e \leq t \text{ and } t_e + W_e + v_{ej} \geq t\}}, \quad \forall j \in J, t \in T. \tag{3.6}$$

At this tactical level, it is not straightforward to determine whether emergency call $e \in E$ is covered according to coverage requirement $k \in K$, because this depends on the assigned waiting time W_e and the varying travel time. To determine the coverage, we introduce a variable o_{ejk} denoting the time required to serve emergency call $e \in E$ according to coverage requirement $k \in K$ when assigned to ambulance base location $j \in J$. For example, when an ambulance must arrive at the incident location within 13 minutes, we have to consider the travel time from the assigned base location $j \in J$ to the location l_e of emergency call $e \in E$. Emergency call $e \in E$ is then covered according to coverage requirement $k \in K$ when $W_e + o_{ejk}$ is less than c_k, whereby c_k represents the time limit for coverage requirement $k \in K$. Again, binary variables Y_{ke} are used to specify whether emergency call $e \in E$ is covered according to coverage requirement $k \in K$. The correct value of Y_{ke} is determined by the following constraint:

$$W_e + \sum_{j \in J} Z_{ej} o_{ejk} \leq c_k + M(1 - Y_{ke}), \qquad \forall e \in E, k \in K, \tag{3.7}$$

where M is a sufficiently large number. To make sure the correct fraction α_{kr} of all emergency calls in region $r \in R_k$ is covered according to coverage requirement $k \in K$, we get the following constraint:

$$\sum_{e \in E} Y_{ke} \mathbb{1}_{\{l_e \in I_{kr}\}} \geq \alpha_{kr} \sum_{e \in E} \mathbb{1}_{\{l_e \in I_{kr}\}}, \qquad \forall k \in K, r \in R_k. \tag{3.8}$$

The objective to minimize the number of ambulances needed is included as follows:

$$\min \sum_{j \in J} \max_{t \in T} A_{jt}. \tag{3.9}$$

Summarizing, the complete formulation of the problem at the tactical level is given by:

$$\begin{aligned}
\min \quad & \sum_{j \in J} \max_{t \in T} A_{jt} && && (3.10)\\
\text{s. t.} \quad & \sum_{j \in J} Z_{ej} = 1, && \forall e \in E, \\
& A_{jt} = \sum_{e \in E} Z_{ej} \mathbb{1}_{\{t_e + W_e \leq t \text{ and } t_e + W_e + v_{ej} \geq t\}}, && \forall j \in J, t \in T, \\
& W_e + \sum_{j \in J} Z_{ej} o_{ejk} \leq c_k + M(1 - Y_{ke}), && \forall e \in E, k \in K, \\
& \sum_{e \in E} Y_{ke} \mathbb{1}_{\{l_e \in I_{kr}\}} \geq \alpha_{kr} \sum_{e \in E} \mathbb{1}_{\{l_e \in I_{kr}\}}, && \forall k \in K, r \in R_k, \\
& X_j, Y_{ki} \in \{0, 1\}, && \forall i \in I, j \in J, k \in K.
\end{aligned}$$

The formulations introduced in this section cannot be used in their present form to solve the problems on the tactical and strategic level simultaneously or separately in reasonable computation time. Therefore, some modifications and simplifications are introduced in the next section such that the problems can be solved within a reasonable amount of time.

3.4 Solution methods

Solving the problem in two stages as described in the previous section may result in a suboptimal solution. However, when combining the strategic and tactical level,

the size of the input instance increases and extra simplifications of the problem might be needed. Therefore, in the following we first present a possible solution approach for solving both levels simultaneously, and after that, present a solution approach for solving the two described problems hierarchically.

3.4.1 Strategic and tactical level combined

As we want to take the uncertainty of emergencies into account, we chose stochastic programming to model the overall problem. A stochastic program in general is a mathematical program where stochastic elements are present in the data, which can influence the objective and/or the constraints. In practice, the detail in which uncertainty is implemented in the model can range from a few scenarios (as the possible outcomes of the data) to specific and precise joint probability distributions.

The formulation presented in this section is a two-stage stochastic program, with we further refer to as SPAB (Stochastic Planning of Ambulances and Bases), using scenarios that resulted from simulating a stochastic arrival process. We chose to model the occurrences of emergencies as a Poisson arrival process and simulated the arrival of emergencies at defined locations within a defined time interval. To simplify the problem we suggest one hour time intervals for the generation of scenarios as an ambulance is often occupied for about one hour when serving an emergency. This means that we do not have to incorporate time into our developed model as all ambulances are assigned to only one emergency. Therefore, we do not have to determine how long an ambulance is occupied.

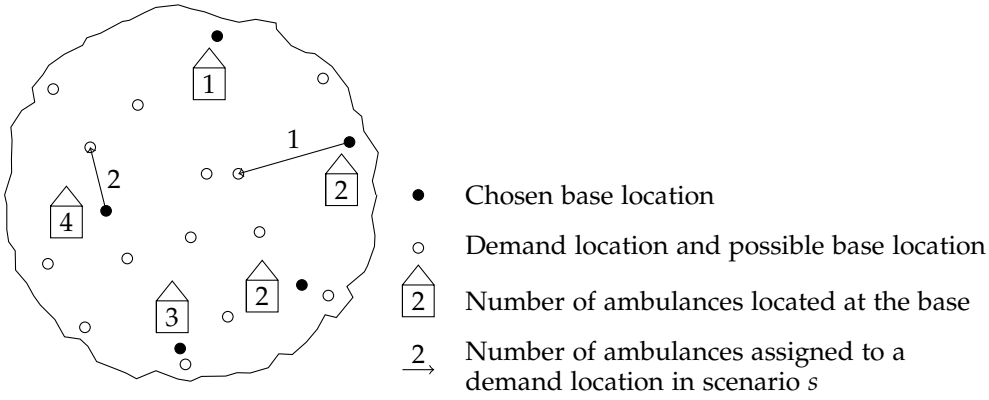


Figure 3.4: The locations of the bases and the number of ambulances are determined simultaneously; for each scenario ambulances are assigned from the bases to the emergency calls

We consider two decision stages in our model. At the first stage, we decide where to locate the ambulance bases and how many ambulances to locate at which

ambulance base. In a second stage, we consider all the potential scenarios and decide which ambulance shall be allocated to which emergency, in order to compute the coverage resulting from the chosen location decisions and ambulance configuration. As a result, we combine the strategic and the tactical level and solve both levels as only one problem as shown in Figure 3.4.

Additionally to the formulations of the previous section, we need a set S of scenarios. Each scenario $s \in S$ occurs with a given probability p^s. As stochastic parameters we have the number of emergencies occurring in demand location $i \in I$; for scenario $s \in S$ this number is denoted by d_i^s. Furthermore, we introduce decision variables G_j representing the number of ambulances at base location $j \in J$ and B_{ij}^s denoting the number of ambulances at base location $j \in J$ that are allocated to emergencies in demand location $i \in I$ when considering scenario $s \in S$. Recall that the variables X_j state whether base location $j \in J$ is opened. This leads to the following formulation, where M denotes a sufficiently large number and C being a constant greater than zero.

$$\min \quad C \cdot \sum_{j \in J} G_j + \sum_{j \in J} X_j \tag{3.11}$$

$$\text{s. t.} \quad \sum_{i \in I} B_{ij}^s \leq G_j, \qquad \forall j \in J, s \in S, \tag{3.12}$$

$$\sum_{j \in J} B_{ij}^s \leq d_i^s, \qquad \forall i \in I, s \in S, \tag{3.13}$$

$$\sum_{i \in I_{kr}} \sum_{j \in J_{ik}} \sum_{s \in S} p^s B_{ij}^s \geq \alpha_{kr} \sum_{i \in I_{kr}} \sum_{s \in S} p^s d_i^s, \qquad \forall k \in K, r \in R_k, \tag{3.14}$$

$$M \cdot X_j \geq G_j, \qquad \forall j \in J, \tag{3.15}$$

$$G_j \in \mathbb{N}, \qquad \forall j \in J, \tag{3.16}$$

$$X_j \in \{0, 1\}, \qquad \forall j \in J, \tag{3.17}$$

$$B_{ij}^s \in \mathbb{N}, \qquad \forall i \in I, j \in J, s \in S. \tag{3.18}$$

The objective function (3.11) minimizes a weighted combination of the number of located ambulances and the number of bases opened. We cannot assign more ambulances from one node to emergencies than we have allocated to the node which is expressed by constraints (3.12). In addition, we are not allowed to assign more ambulances than needed for covering the emergencies at a node j which is stated by constraints (3.13). The constraints (3.14) assure that a fraction α_{kr} of the emergencies in region $r \in R_k$ is definitely covered by ambulances. Finally, the last constraints (3.15) only enable locating ambulances at base location $j \in J$ if the base is opened. Besides, the decision variables must be integer or binary.

When compared to the introduction of the problem in Section 3.3.2, we do not include waiting times and the stochastic driving times. As we use hourly intervals, an ambulance can only serve one call within this hour, and therefore, an ambulance is not used to serve multiple calls. Because of this, there is no waiting time for the

arriving calls. In addition, we do not include stochastic driving times, but only use a fixed driving time which is used to determine whether demand location $i \in I$ is covered or not by one of the chosen base locations. Beforehand, a confidence interval for the driving times can be determined such that, for example, the driving time is met in 95% of the cases.

The approach proposed in this section solves the strategic and tactical level for the ambulance planning problem simultaneously. However, when considering the problem from a practical point of view it might also be natural to solve the problem in two stages. In the next sections, we discuss solution approaches for the strategic and tactical level that can be used to solve the two levels in two steps.

3.4.2 Strategic level

In this section, we introduce a solution approach to solve the strategic level, further referred to as SAP (Strategic Ambulance Planning), without considering the tactical level at the same time. The Integer Linear Program (ILP) (3.4) formulated in Section 3.3.1 can directly be used to solve the strategic level and can be handled by a standard solver. The only question remaining is how to model the demand used in the coverage constraints for each demand location. As the demand fluctuates per day, it is hard to define a fixed value which results in a feasible solution every day. However, the coverage requirements usually only have to hold per year instead of each day, and therefore, it often is sufficient to include the average demand. This way it is ensured that most demand locations can be reached within the given time limit and further alterations can be made on the tactical level by determining the correct number of ambulances per base.

Figure 3.5: Solution for average demand

Figure 3.6: Solution for equal demand

Because the base locations are chosen for a longer period, for example five years, we believe that for the strategic level it is sufficient to only consider the average demand for each demand location $i \in I$. The fluctuations of the demand can then be accounted for at the tactical or operational level. When the average demand is used, the base locations are situated such that demand locations with high demand are covered and locations with less demand might not be covered. However, the coverage requirements ensure that these differences are not too large. An alternative approach would be to set all demand equal to one (while keeping the α-values). In this way, each demand location is equally important. Figures 3.5 and 3.6 show that applying these two possibilities for the demand can result in different solutions. Using the Netherlands as an example and generating the two scenarios, we see that different base locations are chosen. The chosen base locations are depicted by the black squares and the small black dots are the demand locations. When all demand is set to one, 84 bases are needed, while only 81 bases are opened when the average demand is considered. A drawback when setting all demand to one, is that high demand locations might not be covered which leads to one or more of the coverage requirements not being fulfilled in practice. Therefore, we choose the average demand as input for the ILP at the strategic level.

3.4.3 Tactical level

The formulation introduced in Section 3.3.2 cannot be used immediately to solve the problem at the tactical level. In practice, the emergency calls are not known beforehand and the number of emergency calls is too large to be included in an IP. We choose to model the arrival of emergency calls as a stochastic arrival process, and therefore, to solve this problem by means of simulation. The stochastic arrival process is modeled by the random demand scenarios S as defined in the stochastic programming approach. The solution approach developed in this section is further referred to as TAP (Tactical Ambulance Planning).

At the tactical level, we determine the number of ambulances needed at each opened base location based on the opening decisions on the strategic level to fulfill the considered coverage constraints. By using simulation, we can only determine whether a given configuration of the number of ambulances satisfies the given coverage constraints. To minimize the total number of ambulances needed, we combine the simulation with a local search heuristic.

Due to the stochastic arrival process of the emergency calls, the demand at the demand locations fluctuates per time interval. Therefore, the solution obtained by the solution approach used at the strategic level might not lead to a feasible solution in practice. Thus, it might be necessary to open one or more additional base locations to make sure that the coverage requirements are fulfilled. In practice, the ambulance base locations opened in the solution of the strategic level may be assumed to be used for several years. The base locations opened at the tactical level may not be real base locations in that sense, but, for example, a parking lot where the ambulance waits until a call arrives. This means that these base locations may

change over the years.

Algorithm 1 Add bases to fulfill coverage requirements

for all $k \in K$ **do**
  for all $r \in R_k$ with $\sum_{e \in E} Y_{ke} \mathbb{1}_{\{l_e \in I_{kr}\}} < \alpha_{kr} \sum_{e \in E} \mathbb{1}_{\{l_e \in I_{kr}\}}$ **do**
    repeat
      for all $j \in \bigcup_{l_e \in I_{kr}} J_{l_e k}$ **do**
        $\bar{X}_j := 1$
        $\bar{Y}_{kl_e} := \max_{j \in J_{kl_e}} \bar{X}_j$
        $\Delta_j := \frac{\sum_{e \in E} \bar{Y}_{ke} \mathbb{1}_{\{l_e \in I_{kr}\}}}{\sum_{e \in E} \mathbb{1}_{\{l_e \in I_{kr}\}}} - \frac{\sum_{e \in E} Y_{ke} \mathbb{1}_{\{l_e \in I_{kr}\}}}{\sum_{e \in E} \mathbb{1}_{\{l_e \in I_{kr}\}}}$
      end for
      Add base location j for which Δ_j is maximum, i.e., $X_j := 1$ for ambulance base location $\arg\max_{j \in J} \Delta_j$.
      $Y_{kl_e} := \max_{j \in J_{kl_e}} \bar{X}_j$
    until $\sum_{e \in E} Y_{ke} \mathbb{1}_{\{l_e \in I_{kr}\}} \geq \alpha_{kr} \sum_{e \in E} \mathbb{1}_{\{l_e \in I_{kr}\}}$
  end for
end for

Thus, in the first step in our simulation approach we start with the bases opened at the strategic level and we aim at opening one or more extra bases to make the solution robust for fluctuating demand (see Algorithm 1). In this step, we do not yet consider ambulances as first enough bases must be opened such that the coverage requirements given by constraints (3.2) are fulfilled for the considered scenarios. The addition of one or more ambulance bases is done using a greedy strategy by considering the coverage requirements one by one. The coverage requirements can be sorted in an hierarchical way by the number of regions considered. The coverage requirement with the maximum number of regions, i.e., with maximal $|R_k|$ is considered to be the lowest coverage level. The coverage requirements on lower levels are usually stricter and fulfilling these requirements will often already improve the coverage on higher levels. Therefore, we start at the lowest coverage level and check for each region $r \in R_k$ whether the coverage requirements are met or not with the additional chosen bases for the considered scenarios. When for region $r \in R_k$ the coverage requirement is not fulfilled, we add ambulance bases in the following way until the requirement is fulfilled. For each possible base location able to serve demand in this region $r \in R_k$, i.e., ambulance base locations $j \in \bigcup_{i \in I_{kr}} J_{ik}$, we determine what the change in coverage is when this base is opened. Then, we open the base which results in the highest increase in coverage. This is repeated until the coverage requirements are met for this region $r \in R_k$. This procedure is repeated for each coverage level until all coverage requirements are fulfilled for the considered demand scenarios.

After several ambulance base locations are added such that all coverage requirements are fulfilled, a local search procedure is used to minimize the number of ambulances. The first step in this local search heuristic is to determine an initial solution (see Algorithm 2) which determines a sufficient number of ambulances per base. To determine this initial number of ambulances per base, we assign each call to one of the opened bases in the set of opened bases that fulfill all coverage requirements for the considered call location. In this way, each call has its own ambulance. To be more precise, we solve a simple ILP which assigns all generated calls $e \in E$ to one of the opened bases $j \in \cap_{k \in K} J_{l_e k}$ that fulfills all coverage requirements if $\cap_{k \in K} J_{l_e k} \neq \emptyset$. If $\cap_{k \in K} J_{l_e k} = \emptyset$, then the call is assigned to one of the opened bases such that the total number of ambulances needed to serve all calls is minimized. Note that although we consider several scenarios, each call $e \in E$ occurs only in one of the scenarios $s \in S$. Therefore, we can use binary variables B^s_{ej} to assign each call to one of the opened bases. Note that the number of ambulances G_j needed at base $j \in J$ equals $\max_{s \in S} \sum_{e \in E} B^s_{ej}$. As the emergency calls occur at different times in the considered time interval, the number of ambulances needed might be less as multiple calls might be served by the same ambulance. This is the case when there exist two or more emergency calls for which the busy time periods of the assigned ambulances do not overlap. By determining for each time $t \in T$ how many ambulances A_{jt} assigned to base $j \in J$ were simultaneously busy at time $t \in T$, the initial number of ambulances needed might be slightly reduced.

Algorithm 2 Determine starting solution local search

Solve the following ILP:

$$
\begin{aligned}
\min \quad & \sum_{j \in J} G_j \\
\text{s. t.} \quad & \sum_{j \in J} B^s_{ej} = 1, && \forall e \in E, \\
& \sum_{j \in \cap_{k \in K} J_{l_e k}} B^s_{ej} = 1, && \forall e \in E \text{ for which } \cap_{k \in K} J_{l_e k} \neq \emptyset, \\
& G_j \geq \sum_{e \in E} B^s_{ej}, && \forall s \in S, j \in J, \\
& B^s_{ej} \in \{0,1\}, && \forall e \in E, j \in J, s \in S, \\
& G_j \in \mathbb{N}, && \forall j \in J.
\end{aligned}
$$

$G_j := 0$

for all $s \in S$ **do**

$$A_{jt} := \sum_{e \in E} B^s_{ej} \mathbb{1}_{\{t_e \leq t \text{ and } t_e + v_{ej} \geq t\}}$$

$$G_j := \max\left(G_j, \max_{t \in T} A_{jt}\right)$$

end for

In the simple approach sketched above, each call is served directly by one of the bases that fulfill the coverage requirements. It is expected that by adding some

more intelligence in this choice, the number of ambulances needed per base can most likely be reduced. As a call might still be covered according to the coverage requirements when the patient waits until an occupied ambulance becomes available, we may not have to add an additional ambulance for such a call. Thus, we may further improve the initial solution in the next step by a local search procedure.

Algorithm 3 Local search to minimize number of ambulances needed

$F_j := \min\{G_j, 1\}$
repeat
  for all $j \in J$ for which $F_j = 1$ and $G_j > 0$ **do**
    $G_j := G_j - 1$
    Algorithm 4: Reassign calls
    if There exist a $k \in K$ and $r \in R_k$ for which $\sum_{e \in E} Y_{ke} \mathbb{1}_{\{l_e \in I_{kr}\}} < \alpha_{kr} \sum_{e \in E} \mathbb{1}_{\{l_e \in I_{kr}\}}$
    then
      $F_j := 0$
    else
      $\Delta_j := \max_{k \in K} \frac{\sum_{e \in E} \bar{Y}_{ke} \mathbb{1}_{\{l_e \in I_{kr}\}}}{\sum_{e \in E} \mathbb{1}_{\{l_e \in I_{kr}\}}} - \frac{\sum_{e \in E} Y_{ke} \mathbb{1}_{\{l_e \in I_{kr}\}}}{\sum_{e \in E} \mathbb{1}_{\{l_e \in I_{kr}\}}}$
    end if
    $G_j := G_j + 1$
  end for
  $\bar{j} = \arg \min_{j \in J | F_j = 1} \Delta_j$
  $G_{\bar{j}} := G_{\bar{j}} - 1$
  Algorithm 4: Reassign calls
until $\sum_{j \in J} F_j := 0$

As we want to minimize the total number of ambulances needed, i.e., $\sum_{j \in J} G_j$, we reduce the number of ambulances until a further reduction leads to unfulfilled coverage requirements (see Algorithm 3). We do this by reducing for each opened base $j \in J$ the number of ambulances G_j available by one and determining for which of the bases this reduction leads to the least decrease in coverage. For this case, the number of ambulances then definitely is reduced by one. By this approach, we aim to reduce the total number of ambulances as much as possible. Note that we only consider the bases for which the number of available ambulances G_j is at least one and for which the coverage requirements are still fulfilled after the reduction. To incorporate this in our local search approach, we introduce binary variables F_j which are one when base $j \in J$ can be considered for reduction and zero otherwise. For the base with the minimum reduction in coverage, the number of available ambulances G_j is then permanently reduced by one for the remainder of the local search approach. Note that after the reduction, not all calls can be served directly by the nearest base because not enough ambulances are

Algorithm 4 Reassign calls

$W_e := 0$ for all $e \in \bar{E}$.
$A_{jt} := 0$ for all $t \in T$ and the considered base $j \in J$.
for all $t \in T$ **do**
  for all $e \in \bar{E}$ for which $t_e + W_e = t$ **do**
    $\bar{J} := \{j \in \cap_{k \in K} J_{l_e k} | A_{jt} < G_j \text{ for } t \in [t_e + W_e, t_e + W_e + v_{ej}]\}$
    Add nearest base for $e \in E$ to set $\bar{J}$ if $A_{jt} < G_j$ for $t \in [t_e + W_e, t_e + W_e + v_{ej}]$ holds.
    if $\bar{J} = \emptyset$ **then**
      $W_e := W_e + 1$
    else
      Assign call $e \in E$ to nearest base $\bar{j} \in \bar{J}$, i.e., $Z_{e\bar{j}} := 1$.
      $A_{\bar{j}t} := A_{\bar{j}t} + 1$ for $t \in [t_e + W_e, t_e + W_e + v_{ej}]$.
    end if
  end for
end for

available. Therefore, we possibly have to reassign the emergency calls that were assigned to the considered base $j \in J$ (see Algorithm 4). As not always an ambulance is available at the moment of arrival of the call, the call has to wait until an ambulance becomes available at the nearest base or is assigned to a base further away. By only reassigning the emergency calls assigned to the considered base, we give preference to the other calls as these are served immediately by the nearest base. This means that the reassigned calls have to wait until an ambulance becomes available for the entire duration of the transfer of the patient. A better solution can be achieved when all emergency calls are reassigned to a base, however, this will take too much time. The process of reducing and reassigning is repeated until none of the opened bases $j \in J$ is available any more for reduction, i.e., until $F_j = 0$ for all $j \in J$.

3.5 Computational results

In this section, we test the developed approaches on data of the Netherlands. We determine values for the used parameters for both approaches and investigate which method performs the best. In addition, we determine the influence of the input data size on the computation time for both approaches.

3.5.1 Data

The discussed approaches are tested on real-world data of the Netherlands. The considered instances are based on data that is accessible on the internet. As the set of potential bases, we took the set of all 215 bases currently used in the Netherlands.

The locations of these ambulance bases are given by the four digit zip code used in the Netherlands. As the set of hospital locations, we used all hospitals in the Netherlands that have an emergency department. These locations are also specified by a four digit zip code. The set of demand locations consists of all existing four digit zip codes in the Netherlands. Each four digit zip code can be linked to an RD-coordinate which is the coordinate system used in the Netherlands. These RD-coordinates can be used to determine the Euclidean distance between two locations. The Netherlands is divided into 24 regions and for each of these regions the total demand for emergency calls is known. This total demand per region is divided over all demand locations in the region proportional to the number of inhabitants at the zip code of the considered demand location. To generate incoming calls, we assumed that the emergency calls can be modeled as a Poisson process.

In the Netherlands, an ambulance should arrive at the incident location within 13 minutes after the call came in for 97% of all incidents in a region (see [58]). In addition, all patients should arrive at the hospital within 45 minutes after the emergency call came in (see [72]). However, this last coverage requirement is not feasible for the used data as for some demand locations the nearest hospital is more than 45 minutes away. Therefore, we have introduce a coverage percentage of 99.5% for this constraint to guarantee that a feasible solution exists as we did not want to add additional possible base locations without having any information of where they could be located.

To generate scenarios, we generate incoming calls during one hour and do this for 10, 25, 50, 100, 250, 500, and 1000 independent hours. When more hours are considered the solution becomes more reliable, but also the computation time increases. Therefore, we investigate how many hours of incoming calls are needed to achieve a reliable solution within a reasonable amount of time. The generated scenarios are both used for the stochastic program and as input for the model on the tactical level. In the stochastic program, all scenarios have the same probability. The bases opened by the model at the strategic level are used as a starting point for the tactical level.

To determine the influence of the size of the input instance on the computation time, we also consider subinstances with only one, three, five, or ten of the 24 regions. For the objective function of the stochastic program we take $C = 1$.

3.5.2 Results

The results were generated on a PC with an AMD Phenom(tm) II X6 1100T Processor with 3.31 GHz and 16 GB RAM. All the approaches were implemented in AIMMS and the strategic and the stochastic models were solved with CPLEX 12.4. First of all, we investigate the number of scenarios needed (10, 25, 50, 100, 250, 500, and 1000 hours) to achieve reliable solutions within a reasonable amount of time. After this, we compare the two approaches of solving the strategic and tactical level combined and separately. The approach for the strategic and tactical level combined is SPAB and the approach for solving the strategic and tactical sep-

arately is a combination of SAP and TAP which is further referred to as STAP. For the achieved solutions, we give the number of bases opened, the number of ambulances needed and the required computation time for applying the approaches to the considered instances. In addition, we created smaller instances with only one, three, five, and ten regions to be able to analyze the computation times depending on the problem sizes. Due to an increasing computation time some of the instances were stopped after 24 hours. For the instances which were not solved after 24 hours, we give the integrality gap of the ILP for the strategic level.

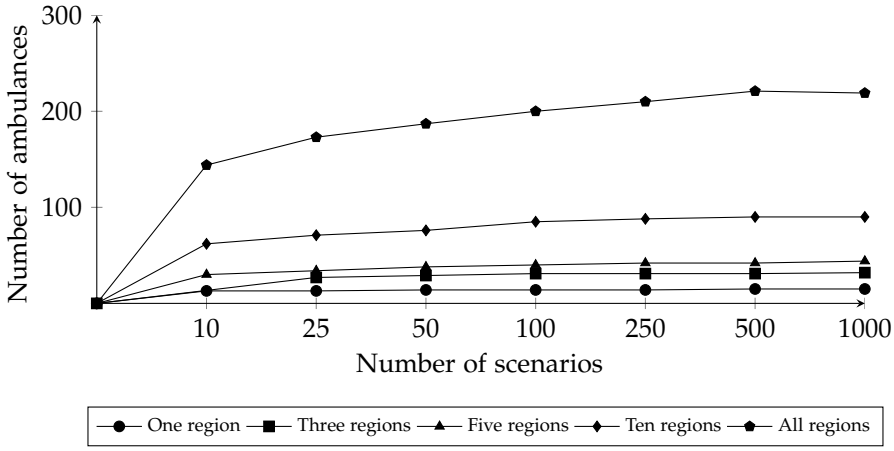


Figure 3.7: SPAB: number of ambulances given for different number of regions for different number of scenarios

First, we aim to determine the number of scenarios that have to be considered such that a reliable solution is found within a reasonable amount of time. To determine this, we first provide graphs which show the number of ambulances needed in the solutions of SPAB and STAP depending on the number of scenarios used. Figures 3.7 and 3.8 show that the number of ambulances increases when the number of scenarios increases, but stabilizes at a certain point. This is to be expected as the number of ambulances needed at a certain base is closely related to the maximum number of calls in the area covered by this bases over all scenarios. This maximum is likely to increase when more scenarios are considered, because a larger range of values is generated. However, at some point the maximum value is achieved and the number of ambulances needed stabilizes.

Figures 3.7 and 3.8 also show that more scenarios are needed when the instance size increases. The total number of ambulances needed stabilizes when the maximum demand is achieved for each demand location in one of the considered scenarios. This is because the configuration for the number of ambulances per base determined by STAP or SPAB has to be feasible for all considered scenarios. When the instance size increases, more demand locations are considered, and thus, more scenarios are needed to achieve this maximum for each demand location.

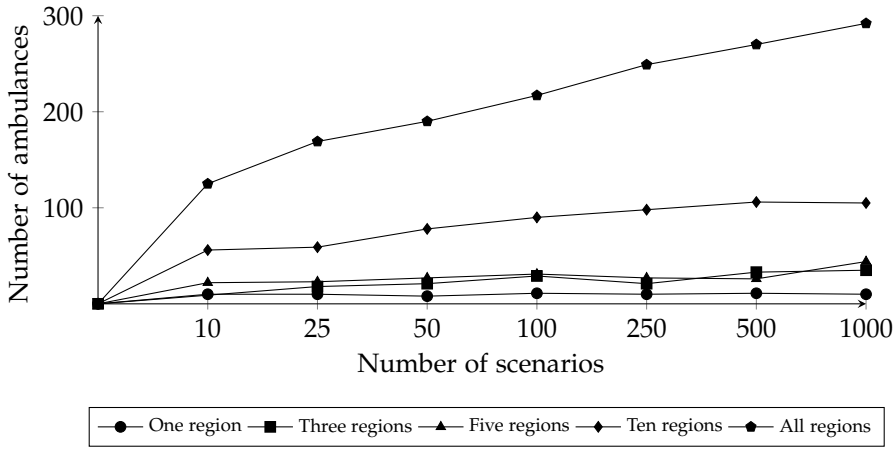


Figure 3.8: STAP: number of ambulances given for different number of regions for different number of scenarios

However, not only the reliability of the solution is important when determining the base locations and number of ambulances needed but also the computation time. Figure 3.9 shows the computation times for STAP and SPAB when 10, 25, 50, 100, 250, 500, and 1000 scenarios are considered for the instance with ten regions. From this graph, we can conclude that the computation time for SPAB increases exponentially whereas the computation time for STAP follows a more linear trend. Based on the results for the computation time and the solution reliability, we suggest to use 100 scenarios to get a good trade-off.

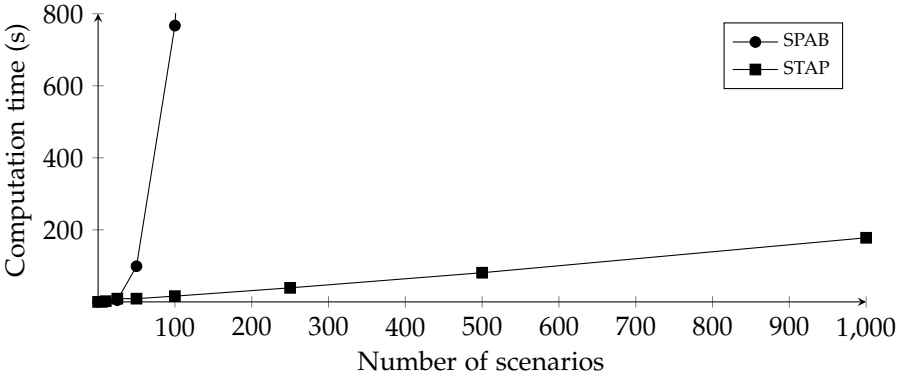


Figure 3.9: Computation time for different number of scenarios and ten regions

To compare SPAB and STAP, we investigate the computation time and solution quality of both approaches. Table 3.1 presents the number of opened bases and

assigned ambulances and Figure 3.10 shows the computation time for the considered instances when 100 scenarios are taken into account. Table 3.1 shows that for almost all instances additional bases are opened at the tactical level to be able to fulfill the coverage requirements for all scenarios. In addition, the number of opened bases for SPAB is higher than for STAP. An explanation for this is that for STAP more emphasis is put on minimizing the number of bases as this is done in the first step, while for SPAB minimizing the number of bases and ambulances is equally important. For the instances with one, three, and five regions the number of needed ambulances is lower for STAP than for SPAB. This is because in STAP an ambulance can be used for more than one emergency call whereas in SPAB only one call is assigned to each ambulance. However, for the instances with 10 and 24 regions, the number of ambulances needed is lower for SPAB than for STAP. For these larger instances, the area covered by bases not near the boarders is larger. The calls in this area can easily be spread out over multiple bases as the locations of these calls are surrounded by bases, which is not the case in boarders regions. By spreading the calls over multiple bases, the number of ambulances needed can be reduced because the peak load on one base is spread out over other opened bases. Therefore, this effect is larger for SPAB than for STAP as more bases are opened in the solution of SPAB.

	# Regions	SAP	TAP	SPAB
# Bases	1	5	5	7
# Ambulances	1	–	11	14
# Bases	3	9	11	13
# Ambulances	3	–	29	31
# Bases	5	11	11	16
# Ambulances	5	–	31	40
# Bases	10	30	33	38
# Ambulances	10	–	90	85
# Bases	24	81	85	108
# Ambulances	24	–	217	200

Table 3.1: Comparison of solving STAP and SPAB (100 scenarios)

Figure 3.10 shows the computation time for SPAB and STAP for the instances with one, three, five, and ten regions when 100 scenarios are evaluated. It is evident that the computation time increases when the input size increases, however, the computation time of SPAB increases exponentially whereas the computation time of STAP follows a more linear trend. This linear trend for STAP does not continue for the instance with all 24 regions because the computation time for SAP takes more than 24 hours to prove that the found solution is optimal. However,

this computation time can be reduced by interrupting the solver when a certain acceptable integrality gap is achieved.

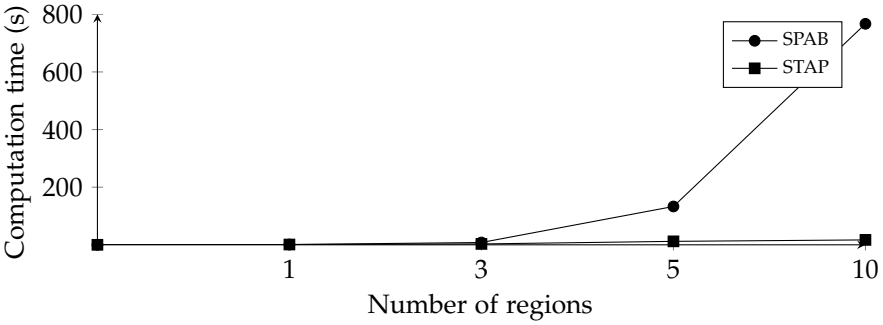


Figure 3.10: Computation time for different number of regions with 100 scenarios

Concluding, when looking at the solution quality, STAP performs better in minimizing the number of bases and SPAB in minimizing the number of ambulances. With SPAB it is also easier to make a good trade-off between the two objectives because of the weighted objective function. However, the computation time for SPAB explodes faster than the computation time of STAP.

Value of C	# Base locations	# Ambulances	Computational time (s)
1	38	85	767
1000	46	80	48

Table 3.2: The impact of C, having ten regions and 100 scenarios

As mentioned above, we have chosen $C = 1$ for the objective function of SPAB. In this case, minimizing the numbers of bases and ambulances is equally important. The results depicted in Table 3.2 show that by choosing a higher value for C the number of ambulances can be reduced by approximately 6%. However, this is at the cost of the number of bases opened as this increases with approximately 20%. Depending on the real-world situation, it can be the case that this is actually preferred. For example, if a large number of bases already exists or additional locations like parking lots can be used, it might be more important to avoid additional ambulances that would cause investments or need extra staff. Surprisingly, Table 3.2 also shows that the computation time can be significantly reduced when a higher value for C is chosen. For example, when considering the instance with ten regions and 100 scenarios, the computation times differ by nearly a factor of 16. Because the emphasis is on the number of ambulances, the options for opening bases are limited, and therefore, the solution space reduces significantly. When C is set to one, a trade-off has to be made between both objectives and this results in

a larger solution space.

3.6 Conclusions and recommendations

In this chapter, we have considered the ambulance planning problem and discussed that it is implicitly treated on different levels. For the first two levels, the strategic and the tactical level, we have presented formulations and solution approaches. Furthermore, for the tactical level we have proposed a simulation combined with a local search. In addition, we gave an approach for solving both levels simultaneously by the means of stochastic programming. Using test instances derived for the Netherlands, we compared the presented approaches.

One main advantage of the formulations proposed in this chapter is that the coverage constraints are modeled in a very generic way, and therefore, they can be easily adapted to different requirements within different countries. As a consequence, the formulations are not only suitable for the Netherlands but probably for the EMS systems of many countries. Based on the first results, it can be stated that the approaches are promising. However, there is still some further work that needs to be done. First of all, the approaches should be tested using real-world data. We are planning to do this in the form of a case study for the Netherlands. During this case study it would also be interesting to compare the current situation to the best solution found by our approach and check to which extent we are able to improve it. A further topic of research is to have a closer look at the different EMS systems in Europe (or even worldwide) to check how good the presented approaches fit to those systems and what further improvements are needed to make them even more generic. Going along with it, the tactical approach should be tested for larger instance sizes, i.e., bigger countries, to prove the applicability. For SAP and SPAB modifications should be examined that make the problems solvable for larger instance sizes. In the current approach, we chose a simple local search because it converges quickly to a local optimum. Other approaches like simulated annealing or tabu search might find better solutions, but will probably take longer. Nevertheless, it might be interesting to implement one of those heuristics and to compare computation times and solution qualities.

Concerning the simulation it would be of interest to implement it in a simulation software like AnyLogic to make solutions also ‘visible’. This would especially be helpful for practitioners when using the approaches in practice. The simulation could then be adapted to the operational level, too. In this chapter, we left out the operational level since it differs significantly from the other two. Nevertheless, this level is very important for the quality of the EMS system and should therefore be incorporated into follow-up research. For example, it could be investigated how much the solution quality on the operational level depends on the solutions obtained at the tactical level when for example ambulances can be relocated to different locations throughout the day. For the operational level the number of ambulances needed together with their locations for different times of the day, week, month, and year could be determined and possible allocation strategies could be

tested. One of the major challenges at the operational level will be that for applying approaches in practice solutions are needed in real-time.

Another adaptation that is needed to make the developed approaches more applicable in practice is to implement different driving times for different situations, e.g., rush hours. This would make the subsets J_{ki} dependent of the considered scenario as some bases in subset J_{ki} might not cover demand location $i \in I$ anymore when the driving time increases. We could, for example, consider three different configurations of the driving time (worst, normal, and best) to keep the computation times reasonable. We plan to incorporate this adaptation in our planned case study for the Netherlands.

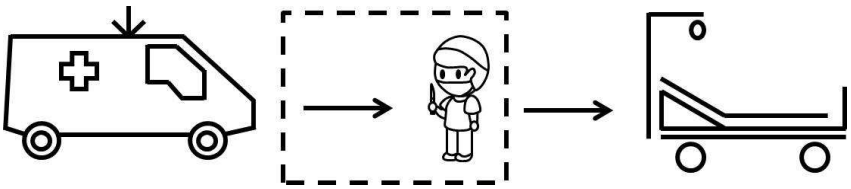
3.7 Appendix

Notation	Description
	Sets
I	set of demand locations
J	set of potential base locations
K	set of coverage requirements
J_{ik}	subset of potential base locations that fulfill coverage requirement $k \in K$ for demand location $i \in I$
R_k	set of regions to be considered for coverage requirement $k \in K$
I_{kr}	subset of demand locations that lie within region $r \in R_k$
E	set of emergency calls
T	set of time points in the considered planning horizon
S	set of scenarios
	Parameters
d_i	demand at demand location $i \in I$
α_{kr}	fraction of demand that should fulfill coverage requirement $k \in K$ for region $r \in R$
t_e	arrival time of emergency call $e \in E$
l_e	location of emergency call $e \in E$
v_{ej}	time an ambulance from base $j \in J$ is occupied when it is assigned to emergency call $e \in E$
o_{ejk}	time emergency call $e \in E$ is occupied according to coverage requirement $k \in K$ when assigned to ambulance base location $j \in J$
c_k	time limit for coverage requirement $k \in K$
p^s	probability for scenario $s \in S$
C	constant greater than zero

Variables	
X_j	binary variable which is one when base location $j \in J$ is selected and zero otherwise
Y_{ki}	binary variable which is one when demand location $i \in I$ is covered according to coverage requirement $k \in k$
Z_{ej}	binary variable which is one when emergency call $e \in E$ is served by ambulance base $j \in J$
W_e	waiting time of emergency call $e \in E$
A_{jt}	number of ambulances at base location $j \in J$ occupied at time $t \in T$
G_j	the number of ambulances at base location $j \in J$
B_{ij}^s	the number of ambulances at base location $j \in J$ allocated to emergencies in demand location $i \in I$ when considering scenario $s \in S$

Part III

Operating room planning



Chapter 4

Minimizing the waiting time for emergency surgery

4.1 Introduction

To maintain a high quality of care it is important to schedule emergency surgeries as quickly as possible. Most hospitals deal with emergency surgeries by reserving some operating room (OR) capacity. This can be done in three ways: (1) dedicating an entire OR to emergency surgeries, (2) scheduling the emergency surgeries in one of the elective ORs, or (3) a combination of (1) and (2). In situation (1), arriving emergency patients are operated immediately if the emergency OR is empty, but if the emergency OR is already occupied, they have to wait until the ongoing surgery has finished. In situation (2), arriving emergency patients can be operated on once an ongoing elective surgery has finished. We call these completion times of the elective surgeries 'break-in-moments' (BIMs). Situation (3) is a combination of situations (1) and (2), which means that the emergency patient is operated immediately if the emergency OR is empty. Otherwise, the patient has to wait until the emergency OR or one of the elective ORs becomes available.

Most literature on OR planning assumes the first situation, i.e., emergency surgeries are performed in a dedicated emergency OR, and therefore, emergency surgeries are not taken into account when planning the elective surgeries. However, Wullink et al. [97] have shown that for a hospital with 12 or more ORs it may be better, in terms of waiting time, staff overtime, and OR utilization, to schedule the emergency surgeries in one of the elective ORs. Gerchak et al. [38] and Lamiri et al. [59] consider this situation on the tactical level and determine which elective surgeries can be scheduled in a given period of time such that enough time is reserved for emergency surgeries. Pham and Klinkert [77] and Dexter et al. [30] consider the problem of scheduling emergency surgeries on the operational on-line level by determining how and in which order the arriving emergency surgeries should be inserted into the elective schedule. However, none of the existing literature considers this problem on the operational off-line level where the arriving emergency surgeries have to be taken into account when sequencing and scheduling the elective surgeries. By doing this, the waiting time of emergency surgeries can be reduced by spreading the BIMs as evenly as possible over the day. This can be achieved by sequencing the surgeries in their assigned OR, such that the maximum interval between two consecutive BIMs is minimized. We refer to this problem as the BIM problem, which is visualized in Figure 4.1.

For the BIM problem, there are two situations. In the first situation, each elective

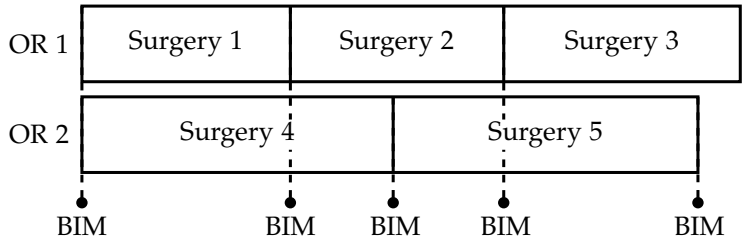


Figure 4.1: Break-in-moments

surgery is preassigned to an OR. In the second situation, the assignment of elective surgeries to an OR is part of the problem. In this research, we consider the first situation, i.e., we focus on sequencing the elective surgeries in their preassigned OR, because in practice, the assignment of elective surgeries to an OR is often fixed due to medical or practical reasons. In addition, including the assignment of elective surgeries to an OR would make the problem even harder to solve, and therefore, we choose to first solve the basic problem in which the surgeries are preassigned to an OR. The methods developed for this problem can be used in future research where the assignment of surgeries to an OR is part of the problem. For the situation considered in this chapter, the selection of surgeries for a specific OR can, for example, be done using the methods proposed by Gerchak et al. [38] and Lamiri et al. [59]. They propose methods to determine which elective surgeries should be scheduled such that overtime is minimized when a random capacity is needed for emergency surgeries.

As far as we know, the BIM problem is a new type of scheduling problem which has not yet been discussed in literature. A problem quite similar to the BIM problem is the Scheduling with Safety Distances problem introduced by Spieksma et al. [84]. The objective of this problem is to schedule the jobs such that the distance between two completion times is larger than a given value d. Therefore, this objective differs from our objective because we aim to minimize the maximum distance between two consecutive BIMs.

In Section 4.2, we give a formal definition of the BIM problem and prove that the BIM problem is strongly $\mathcal{NP}$-hard. We discuss several exact and heuristic solution methods for the BIM problem in Section 4.3. Computational results for these methods are given in Section 4.4.

In practice, the initial schedule is disrupted by emergency surgeries and the stochastic durations of elective surgeries. As a result, the completion times of the elective surgeries, and therefore, the BIMs change, leading also to a change of the maximum distance between two BIMs. To investigate the robustness of the initial schedules, we conduct and report on a simulation study in Section 4.5. Section 4.6 presents conclusions and gives recommendations for further research.

4.2 Problem formulation

In Section 4.2.1, we formally introduce the BIM problem and in Section 4.2.2, we prove that the problem is strongly $\mathcal{NP}$-hard.

4.2.1 Problem description

The BIM problem focuses on sequencing elective surgeries which are assigned to specific ORs. The set of these elective surgeries is given by set $I = \{1, \ldots, M\}$ and each surgery $i \in I$ has an expected duration p_i. We assume that all surgeries have to be scheduled successively with no breaks in between. The set of ORs is given by $J = \{1, \ldots, N\}$ and the set of surgeries that are assigned to OR $j \in J$ is given by $I_j \subset I$. The sets I_j form a partition of set I, and therefore, $\sum_{j \in J} M_j = M$, where M_j denotes the number of surgeries in I_j. Let $O(i)$ denote the OR to which surgery $i \in I$ is assigned.

The start time of OR $j \in J$ is denoted by s_j and e_j denotes the moment that all surgeries in OR $j \in J$ are expected to be completed, i.e., $e_j = s_j + \sum_{i \in I_j} p_i$. The interval for which all operating rooms are occupied is called the 'occupied interval'. The start time of the occupied interval is $s = \max_{j \in J} s_j$ and the end time is given by $e = \min_{j \in J} e_j$.

We define all moments that can be used to start an emergency surgery as a BIM. These moments include the start and end time of the occupied interval, as well as all completion times of surgeries within the occupied interval. To determine the length of the intervals between two BIMs, the completion times of the surgeries must be ordered in non-decreasing order. We define the interval between two subsequent BIMs as a break-in-interval (BII). Figure 4.2 visualizes the occupied interval, the BIMs, and BIIs for a simplified situation involving two ORs.

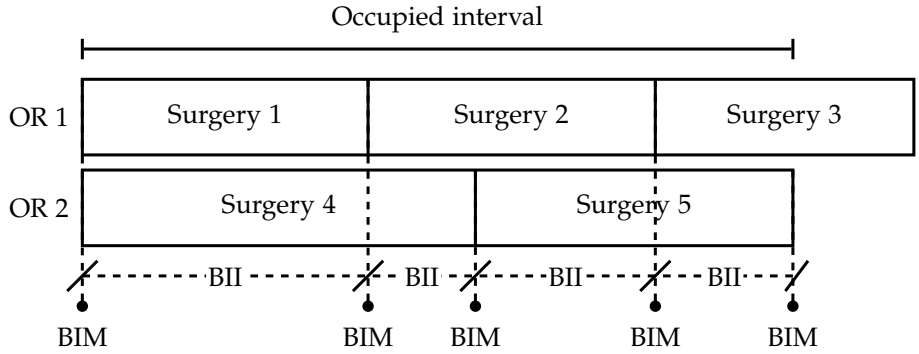


Figure 4.2: BIMs, BIIs, and occupied interval

The timing of the BIMs, and thus the length of the BIIs, depends on the sequence of the elective surgeries in each OR. In practice, BIMs appear to be distributed unevenly over the occupied interval, which results in lengthy BIIs that increase the

expected waiting time for emergency surgery, and therefore, risks medical complications or even mortality. In this research, we aim to minimize the expected waiting time for emergency surgery by sequencing surgeries in such a way that the BIMs are spread as evenly as possible over the day. We choose to measure 'evenly' by minimizing the maximum BII per day, i.e., min max BII.

By considering the ORs as machines and the surgeries as jobs, we see similarities between the BIM problem and machine scheduling problems. However, our objective of spreading the completion times as evenly as possible has, as far as we know, not previously been treated in the literature. Nevertheless, this objective may occur in several other fields in which walk-in customers have to be served as well as elective procedures.

Another application of the BIM problem in the healthcare sector is to spread the workload on the holding and recovery department. This is the department where patients are prepared before surgery and recover after surgery. When the end times, and therefore, also the start times of surgeries are spread as evenly as possible over the day, the arrivals in and departures from the holding and recovery department are also spread over the day. Because patients need the most care at their arrival and departure, the work balance of care for these departments is improved. However, the difference in the length of stay of patients may reduce this positive effect.

4.2.2 Problem complexity

Theorem 4.1. *The BIM problem is strongly $\mathcal{NP}$-hard for two or more operating rooms.*

Proof. We prove the theorem by reducing 3-partition to the BIM problem. The 3-partition problem can be formulated as follows. Given positive integers $a_1, \ldots, a_{3t}$, and b with $\sum_{j=1}^{3t} a_j = tb$, do there exist t pairwise disjoint subsets $R_l \subset \{1, \ldots, 3t\}$ such that $\sum_{j \in R_l} a_j = b$ for $l = 1, \ldots, t$? The 3-partition problem is proven to be strongly $\mathcal{NP}$-hard (see Garey and Johnson [36]).

The reduction is based on the following transformation. Consider two ORs and $5t - 1$ surgeries with the following processing times:

$$\begin{array}{lll} p_i = 4b & \text{for } 1 \leq i \leq t-1, & i \in I_1, \\ p_i = a_{i-t+1} & \text{for } t \leq i \leq 4t-1, & i \in I_1, \\ p_i = 3b & \text{for } 4t \leq i \leq 4t+1, & i \in I_2, \\ p_i = 5b & \text{for } 4t+2 \leq i \leq 5t-1, & i \in I_2. \end{array}$$

First, we notice that the lower bound on our objective min max BII is given by $2b$, since the surgeries of length $4b$ can only be split up by at most one BIM; see Figure 4.3. Based on this, we prove that the BIM problem has a solution with max BII $= 2b$ if and only if there exists a solution to the 3-partition problem.

If the 3-partition problem has a solution, we can create an optimal schedule for the BIM problem with max BII $= 2b$ as shown in Figure 4.3. For the jobs $1, \ldots, t-1$ and $4t, \ldots, 5t-1$, the index and the corresponding processing times are given.

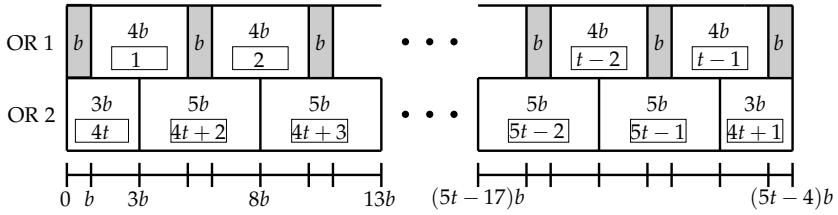


Figure 4.3: Reduction of 3-partition to the BIM problem

The t grey blocks of length b contain the jobs $t, \ldots, 4t-1$ and correspond to the t subsets for which $\sum_{j \in R_l} a_j = b$.

Furthermore, we show that this is the only way in which a solution with $\max \text{BII} = 2b$ can be created.

- There must be a BIM exactly in the middle of each surgery of length $4b$, since otherwise $\max \text{BII}$ is larger than $2b$. Therefore, a surgery of length $5b$ must start or end in the middle of a surgery of length $4b$. This means that there is a gap of length b between two surgeries each of length $4b$.
- Note that the surgeries of length $3b$ can only be placed at the beginning and end of OR 2 to achieve a solution with $\max \text{BII} = 2b$. This results in a gap of length b at the beginning and end of OR 1.
- Combining the first two points results in a set of t gaps of length b, in which t sets of surgeries with $\sum_{j \in R_l} a_j = b$ have to be scheduled in order to obtain a solution to the BIM problem.

□

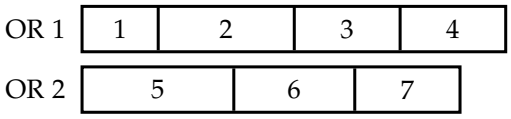
4.3 Solution methods

In this section, we discuss several solution methods for the BIM problem. In Section 4.3.1 we give an Integer Linear Program (ILP), which can be used to solve small instances and to benchmark heuristic solution methods. In Section 4.3.2, we discuss some constructive heuristics, which generate single solutions. Improvement heuristics, which iteratively improve an initial solution, are discussed in Section 4.3.3. In Section 4.3.4, some Shifting Bottlenecks Heuristics (SBH) are discussed.

4.3.1 Exact solution method

To solve the BIM problem to optimality, we may model it as an Integer Linear Program (ILP). Based on this model, small instances can be solved using existing ILP solvers. To define proper decision variables, note that a solution to the BIM problem consists of a sequence of surgeries for each OR. These sequences are

called 'local' sequences. However, in order to determine the BIIs, and thereby the maximum BII, we need to determine the sequence of the BIMs over all ORs, i.e., the completion times C_i of all surgeries $i \in I$. We call this sequence the 'global sequence'. Figure 4.4 shows an example of local and global sequences.



Global sequence: (1,5,2,6,3,7,4)

Figure 4.4: Local sequences and global sequence

The global sequence can be defined by binary variables Y_{ik}, which are one if and only if surgery $i \in I$ is scheduled before surgery $k \in I$ in the global sequence. To ensure that either surgery $i \in I$ is scheduled before surgery $k \in I$, i.e., $Y_{ik} = 1$, or surgery $k \in I$ is scheduled before surgery $i \in I$, i.e., $Y_{ki} = 1$, the following constraints are introduced:

$$Y_{ik} + Y_{ki} = 1, \qquad \forall i, k \in I. \tag{4.1}$$

Based on the values Y_{ik}, the positions of surgeries $i \in I$ in the global sequence are given by:

$$Z_i = 1 + \sum_{k \in K} Y_{ki}, \qquad \forall i \in I. \tag{4.2}$$

To ensure that the chosen Y_{ik} values are consistent, the Z_i's must form a strict totally ordered set, meaning that between each pair of surgeries precisely one precedence relation is chosen and that the corresponding relations are transitive. If this is not the case, the Y_{ik} values may contain cycles leading to inconsistencies whereby a surgery is scheduled before itself. To model this, we must have that $Z_i < Z_k$ if $Y_{ik} = 1$, which is ensured by the following constraints:

$$Z_i \leq Z_k - 1 + \mathcal{M} Y_{ki}, \qquad \forall i, k \in I, \tag{4.3}$$

where $\mathcal{M}$ is a sufficiently large number. Note that this constraint only becomes active when $Y_{ki} = 0$.

A solution to the BIM problem is only feasible, when the completion times resulting from the local sequences are consistent with the global sequence. This means that when surgery $i \in I$ is scheduled before surgery $k \in I$ in the global sequence ($Y_{ik} = 1$), this has to imply that the completion time of surgery $i \in I$, given by C_i, is smaller than or equal to the completion time of surgery $k \in I$, given by C_k. This is ensured by the following constraints:

$$C_i - \mathcal{M}(1 - Y_{ik}) \leq C_k, \quad \forall i, k \in I, \tag{4.4}$$

where C_i is given by the start time of OR $O(i)$ plus the duration of the surgeries scheduled before surgery $i \in I$ in this OR, including the duration of surgery $i \in I$, i.e.:

$$C_i = s_{O(i)} + \sum_{k \in I_{O(i)}} p_k Y_{ki}, \quad \forall i \in I. \tag{4.5}$$

Note that for two surgeries $i, k \in I$ scheduled in the same OR, constraint (4.5) already ensures that the values of C_i, C_k, and Y_{ik} are compatible, i.e., $C_i \leq C_k$ when $Y_{ik} = 1$.

The BIIs are now given by the difference of the completion times of two consecutive surgeries $i \in I$ and $k \in I$ in the global sequence, i.e., by $C_k - C_i$. Surgeries $i \in I$ and $k \in I$ are scheduled consecutively when surgery $i \in I$ is scheduled immediately before surgery $k \in I$ in the global sequence, i.e., when $Z_k - Z_i = 1$. To model this, we introduce binary variables D_{ik}, which take value one if $Z_k - Z_i \leq 1$. This is ensured by constraints (4.6). The variables D_{ik} are used to only include the positive BIIs of consecutive BIMs when determining the maximum BII (see constraint (4.8)):

$$Z_k - Z_i \geq 2 - \mathcal{M} D_{ik}, \qquad \forall i, k \in I. \tag{4.6}$$

Note that $D_{ik} = 1$ if either surgery $k \in I$ is scheduled directly after or somewhere before surgery $i \in I$ in the global sequence. In the latter case, $C_k - C_i$ is negative. However, this will not cause any problems, because the maximum BII is at least zero.

A BIM should only be taken into account when it lies inside the occupied interval. To model this, we introduce binary variables W_i, which have to take value one if the completion time of surgery $i \in I$ is larger than or equal to the start time of the occupied interval, i.e., when $C_i \geq s$. Binary variables V_i have to take value one if the completion time of surgery $i \in I$ is smaller than or equal to the end time of the occupied interval, i.e., when $C_i \leq e$. This is ensured by the following constraints, where ϵ is a sufficiently small number:

$$\begin{aligned} C_i &\geq e + \epsilon - \mathcal{M} V_i, && \forall i \in I, \\ C_i &\leq s - \epsilon + \mathcal{M} W_i, && \forall i \in I. \end{aligned} \tag{4.7}$$

Using the variables D_{ik}, V_i, and W_i, we can now model the objective to minimize the maximum BII by introducing a variable H, which denotes the maximum BII and for which the following constraints must hold:

$$H \geq C_i - C_k - \mathcal{M}(1 - D_{ik}) - \mathcal{M}(1 - V_i) - \mathcal{M}(1 - W_k), \quad \forall i, k \in I. \tag{4.8}$$

This constraint states that H should be greater than or equal to each BII of two consecutive surgeries in the global sequence, when their completion times lie inside the occupied interval. The variable D_{ik} guarantees that the only positive BIIs considered for H are those between two consecutive BIMs (note that when $Z_i - Z_k > 1$, the variable D_{ik} is equal to zero). The variables W_k and V_i guarantee that only BIMs within the occupied interval are considered, because when $C_i > e$ or $C_k < s$, the variable V_i or W_k is equal to zero. The objective function is now given by min H. Summarizing, the resulting ILP for the BIM problem is:

$$\begin{aligned} \min \quad & H & \\ \text{s. t.} \quad & (4.1)\text{-}(4.8), & \\ & Y_{ik}, D_{ik}, V_i, W_i \in \{0,1\}, & \forall i, k \in I, \\ & C_i \in \mathbb{R}, Z_i \in \mathbb{N}, & \forall i \in I. \end{aligned} \tag{4.9}$$

This ILP consists of $2M^2 + 4M + 1$ variables and $4M^2 + 4M$ constraints, where M is the number of surgeries. The $2M^2 + 4M + 1$ variables can be divided into $2M^2 + 2M$ binary variables and $2M + 1$ continuous variables.

In practice, we may need to add more constraints to this ILP. Some surgeries may have to be performed early in the day, for example surgeries on children, while other surgeries may have to be performed at the end of the day, for example when extra cleaning is needed after surgery. We also have to take into account the availability of scarce resources, such as microscopes or X-ray machines. In addition, the surgeon may have a preference for the sequence of surgeries. Many of these constraints can be incorporated in the ILP model in a straightforward way.

4.3.2 Constructive heuristics

As the ILP model of the previous section can only be used to solve small instances, in this section, we discuss four constructive heuristics which can be used to generate a single solution. If there are ORs that start before the occupied interval (i.e., if not all ORs start at the same time), we perform a preprocessing step in which some surgeries are removed. These are the surgeries that are scheduled outside the occupied interval in the preprocessing step, which means that both the start and completion time of the surgery lie before the start time or after the end time of the occupied interval. We give preference to removing lengthy surgeries, because short surgeries may help more to decrease the maximum BII.

Step 1. Schedule the surgeries of each OR by Longest Processing Time first (LPT). Let $I^1 \subset I$ be the set of surgeries for which the completion time C_i in this LPT schedule is smaller than or equal to the start time of the occupied

interval, i.e., for which $C_i \leq s$. Delete these surgeries from set I and I_j and update the start time of the OR, i.e., $I \setminus I^1$, $I_j \setminus I_j^1$, and $s_j = \max_{i \in I_j^1} C_i$.

Step 2. Schedule the remaining surgeries of each OR by Shortest Processing Time first (SPT). Let $I^2 \subset I$ be the set of surgeries for which the start time $C_i - p_i$ in this SPT schedule is greater than or equal to the end time of the occupied interval, i.e., for which $C_i - p_i \geq e$. Delete these surgeries from set I and I_j and update the end time of each OR, i.e., $I \setminus I^2$, $I_j \setminus I_j^2$, and $e_j = \min_{i \in I_j^2} C_i - p_i$.

After this preprocessing step, we have at least one surgery for each OR that can be scheduled first or last such that it partly lies in the occupied interval. However, there still may exist surgeries with a short duration that lie completely outside the occupied interval when scheduled first or last in their OR.

The first heuristic considered is based on list scheduling and specifically on SPT. Furthermore, we propose three other heuristics that aim to sequence the surgeries such that every BII approaches a lower bound λ on the max BII, which is given as follows:

$$\lambda = \frac{e - s}{1 + \sum_{j \in J} (M_j - 1)} = \frac{e - s}{1 + M - N}. \tag{4.10}$$

This lower bound reflects the interval between two BIMs if all completion times are evenly distributed in the occupied interval. The length of the occupied interval, given by $e - s$, is divided by the number of expected BIIs. For each OR the number of expected BIIs equals the number of surgeries assigned to this OR given by M_j. However, in this way, we count the end of the occupied interval N times while we only have to count it once. Therefore, the number of expected BIIs is given by $1 + M - N$. Note that this lower bound λ is the best possible objective in an ideal situation. Therefore, we may use it as a goal value for heuristic approaches. Also note that this bound is one of the reasons to do the preprocessing step, because else $1 + M - N$ surely would be an overestimate of the expected number of BIMs in the occupied interval.

List scheduling

The list scheduling algorithm sorts the surgeries in some order, and then, iteratively, adds surgeries to the schedule in this order. We add a quality constraint, which ensures that the first entry on the list is only added if it does not violate this quality constraint. Otherwise, the next surgery on the list that does not violate the quality constraint is scheduled. If there is no such entry, the last surgery on the list is scheduled. The quality constraint is defined as follows. Let λ be defined as in (4.10) and let τ be the length of the BII that will be created by the considered surgery; see Figure 4.5. The quality constraint is defined as follows:

$$\tau \geq \lambda(1-\beta). \tag{4.11}$$

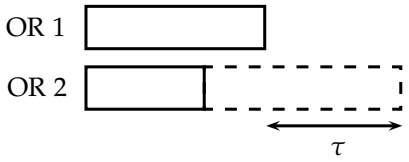


Figure 4.5: Scheduling the next surgery in OR 2 gives a BII of length τ

To spread the BIMs as good as possible, the created BII should not be smaller than λ minus a factor β of λ with $0 < \beta < 1$, i.e., we only accept a new BII if it is relatively close to its goal value λ. Next to the quality constraint, we also guarantee that the BIMs occur in non-decreasing order. This means that we only add a BIM to the set of already achieved BIMs when the new BIM does not split up any of the already achieved BIIs and when after adding this BIM for all ORs another BIM exists which does not split up any of the already achieved BIIs. Thus, we guarantee that by adding this BIM, we can still achieve a non-decreasing order of BIMs in the next iterations.

Within the list scheduling algorithm, every possible order of surgeries may be used. We have chosen the SPT order, which sorts all surgeries in non-decreasing order of processing times, because short surgeries often have a low variation in duration which makes the schedule more robust. The resulting algorithm is called the SPT-algorithm.

Fixed goal values

The 'fixed goal values' constructive heuristic has some similarities to the list scheduling algorithm, which is described above. It also builds a schedule in which the BIIs try to approximate the goal value λ as good as possible. However, now no pre-specified order of the surgeries is given, but in each step, a surgery is selected for which the new created BII approximates λ the best. We start at the beginning of the day and schedule the surgery for which the completion time approximates $s + \lambda$ the best. Next, we choose from the remaining surgeries the surgery for which its completion time best approximates $s + 2\lambda$. We continue until all surgeries are scheduled. As for list scheduling, we again guarantee that the BIMs occur in non-decreasing order. The specific description of this constructive heuristic is given below. Here, B_j is the new start time of OR j resulting from surgeries scheduled in previous iterations.

Step 1. Determine λ, set $t = 1$, $I' = I$, $I'_j = I_j$, and $B_j = s_j$.

Step 2. Determine for each unscheduled surgery $i \in I'$ the completion time if it is scheduled next, i.e., $C_i = B_{O(i)} + p_i$.

Step 3. Let $\bar{I} \subseteq I'$ be the set of unscheduled surgeries $i \in I'$ with $C_i \leq \min_{j \in J}\big(B_j + \max_{k \in I'_j} p_k\big)$. The restriction to the subset $\bar{I}$ is needed to guarantee that the sequence of BIMs is non-decreasing. Set $k = \arg\min_{i \in \bar{I}} |\lambda t - C_i|$.

Step 4. Determine for OR $O(k)$ the new start time, i.e., $B_{O(k)} = B_{O(k)} + p_k$. Delete surgery $k \in I$ from set I' and $I'_{O(k)}$, i.e., $I' = I' \setminus \{k\}$ and $I'_{O(k)} = I'_{O(k)} \setminus \{k\}$. Set $t = t + 1$.

Step 5. If $I' = \emptyset$ then stop. Else go to Step 2.

The goal value λ is calculated once at the beginning of the algorithm. However, it may happen that the scheduled surgeries always under- or overestimate the goal value λ. Therefore, it may be beneficial to let λ depend on the already scheduled surgeries. The following section describes a heuristic that incorporates this suggestion.

Flexible goal values

The idea of 'flexible goal values' is similar to the 'fixed goal values' algorithm. However, in this heuristics, the value of λ is updated in each step. After a surgery is scheduled, we determine the new occupied interval which starts at the completion time of the last scheduled surgery. In addition, the number of surgeries to be scheduled, is decreased by one. Let B denote the new start of the occupied interval and K the number of scheduled surgeries, then:

$$\lambda = \frac{e - B}{1 + M - N - K}. \tag{4.12}$$

The specific description of this constructive heuristic is given by the following steps.

Step 1. Set $I' = I$, $I'_j = I_j$, $B_j = s_j$, and $K = 0$.

Step 2. Set $B = \max_{j \in J} B_j$ and determine λ.

Step 3. Determine for each surgery $i \in I'$ the completion time if it would be scheduled next, i.e., $C_i = B_{O(i)} + p_i$.

Step 4. Let $\bar{I} \subseteq I'$ be the set of unscheduled surgeries $i \in I'$ with $C_i \leq \min_{j \in J}\big(B_j + \max_{k \in I'_j} p_k\big)$. The restriction to the subset $\bar{I}$ is needed to guarantee that the sequence of BIMs is non-decreasing. Set $k = \arg\min_{i \in \bar{I}} |B + \lambda - C_i|$.

Step 5. Determine for OR $O(k)$ the new start time, i.e., $B_{O(k)} = B_{O(k)} + p_k$. Delete surgery $k \in I$ from set I' and $I'_{O(k)}$, i.e., $I' = I' \setminus \{k\}$ and $I'_{O(k)} = I'_{O(k)} \setminus \{k\}$. Set $K = K + 1$.

Step 6. If $I' = \emptyset$ then stop. Else go to Step 2.

It may be worthwhile to apply this heuristic not only in a forward manner but also in a backward manner starting at the end time e of the occupied interval. The following section describes a heuristic that incorporates this suggestion.

Forward-backward scheduling

The 'forward-backward' algorithm builds further on the 'flexible goal values' algorithm. However, in this heuristic, we schedule both forwards and backwards. Forward scheduling is the scheduling of surgeries one after another from the start of the day (s_j) towards the end of the day (e_j) of each OR j, while the reverse holds for backward scheduling. Backward scheduling is possible, since the number and the durations of surgeries in an OR are known. For the 'flexible goal values' algorithm, it may happen that the lengthy surgeries are scheduled at the end of the day, which may result in large BIIs. By scheduling forwards and backward, we try to avoid this. As in the 'flexible goal values' algorithm, the value of λ is updated in each step by adjusting the start and end times of the occupied interval. The adjusted start time of the occupied interval is denoted by B and the adjusted end time of the occupied interval is denoted by F. Thus:

$$\lambda = \frac{F - B}{1 + M - N - K}. \tag{4.13}$$

The specific description of this constructive heuristic is given as follows.

Step 1. Set $I' = I$, $I'_j = I_j$, $B_j = s_j$, $F_j = e_j$, and $K = 0$.

Step 2. Set $B = \max_{j \in J} B_j$, $F = \min_{j \in J} F_j$, and determine λ.

Step 3. Determine for each surgery $i \in I'$ the completion time if it would be scheduled next forward, i.e., $C_i = B_{O(i)} + p_i$, and if it is scheduled backward, i.e., $\tilde{C}_i = F_{O(i)}$.

Step 4. Let $\bar{I}_f \subseteq I'$ be the set of unscheduled surgeries $i \in I'$ with $C_i \leq \min_{j \in J}\big(B_j + \max_{k \in I'_j} p_k\big)$. The restriction to the subset $\bar{I}_f$ is needed to guarantee that the sequence of BIMs is non-decreasing. Let $\delta_f = \min_{i \in \bar{I}} |B + \lambda - C_i|$ and set $k_f = \arg\min_{i \in \bar{I}} |B + \lambda - C_i|$. Let $\bar{I}_b \subseteq I'$ be the set of unscheduled surgeries $i \in I'$ with $\tilde{C}_i - p_i \geq \max_{j \in J}\left(F_j - \max_{k \in I'_j} p_k\right)$. The restriction to the subset $\bar{I}_b$ is needed to guarantee that the sequence of BIMs is non-decreasing. Let $\delta_b = \min_{i \in \bar{I}} |F - \lambda - (\tilde{C}_i - p_i)|$ and set $k_b = \arg\min_{i \in \bar{I}} |F - \lambda - (\tilde{C}_i - p_i)|$.

Step 5. If $\delta_f < \delta_b$, then $k = k_f$, else $k = k_b$. Determine for OR $O(k)$ the new start or end time, i.e., if $\delta_f < \delta_b$, then $B_{O(k)} = B_{O(k)} + p_k$, else $F_{O(k)} = F_{O(k)} - p_k$. Delete surgery $k \in I$ from set I' and $I'_{O(k)}$, i.e., $I' = I' \setminus \{k\}$ and $I'_{O(k)} = I'_{O(k)} \setminus \{k\}$. Set $K = K + 1$.

Step 6. If $I' = \emptyset$ then stop. Else go to Step 2.

4.3.3 Improvement heuristics

The constructive heuristics presented above all have a local view when scheduling the next surgery and do not consider the 'global' consequences of the decisions. Therefore, the resulting schedule may leave room for improvement. To achieve this improvement we use local search procedures. For the local search procedures described in this section, we use a 2-exchange neighborhood. In this neighborhood, a solution is a neighbor when it can be achieved by exchanging two surgeries in the same OR. Note that exchanging surgeries from two different ORs is not allowed. We also experimented with other neighborhood structures [74], but the 2-exchange neighborhood performed the best in preliminary tests. The other neighborhoods can only make a limited change in the solution, and therefore, will not explore the whole solution space. In each iteration of the local search procedure, a solution from the neighborhood is generated and evaluated. The procedure either accepts or rejects this candidate solution as the next solution to move on to, based on a given acceptance-rejection criterion. In the following two sections, we discuss two types of local search procedures, namely Simulated Annealing (SA) and Tabu Search (TS) metaheuristics (see [32]).

Before we go into detail, we first introduce a change in the way the objective function is defined. If we only minimize the maximum BII, two solutions are considered to be equally good if the length of the maximum BIIs are the same. However, the other BIIs also influence the quality of a solution in practice, because spreading the BIMs makes a solution more robust. Therefore, when the maximum BIIs of two solutions are of equal length and the second largest BII of one solution is smaller than the second largest BII of another solution, the first solution is better. Preliminary tests have shown that it is sufficient to consider only the three largest BIIs. Therefore, we introduce a new objective function which is a lexicographic vector of the three largest BIIs. Furthermore, note that the constructive heuristics, due to their description, already focus on all BIIs. We do not include this new objective function in the ILP, because we only use the ILP to determine whether the heuristic solution methods achieve the minimal maximum BII. We do not use the ILP solution in practice, and therefore, it is not necessary to spread the BIMs as evenly as possible.

Simulated annealing

Recall that each step of SA possibly moves from the current solution to a randomly selected neighbor solution. If the neighbor solution has a lower objective function value than the current solution, the neighbor solution is accepted. Otherwise, the neighbor solution is accepted with probability $e^{\frac{-\Delta}{T}}$, where Δ is the difference between the objective value of the current and the neighbor solution and T is the current temperature. If the neighbor solution is rejected, SA stays at the current solution. See Section 2.4.2 for a more detailed description of this method. Summarizing, our implementation of SA is as follows, where $\bar{Q}$ denotes the current best solution:

Step 1. Generate an initial solution Q using some heuristic and determine its objective function vector $f(Q)$. Set $\bar{Q} = Q$. Set an initial temperature T_0 and a reduction factor α. Set $T = T_0$.

Step 2. Repeat L times:

Step (a) Select a neighbor solution Q' of solution Q at random and determine $f(Q')$.

Step (b) If $f(Q') \leq f(Q)$, set $Q = Q'$, and if $f(Q') \leq f(\bar{Q})$, set $\bar{Q} = Q'$. If $f(Q') > f(Q)$, set $Q = Q'$ with probability $e^{\frac{-\Delta}{T}}$.

Step 3. Set $T = \alpha T$. If $T < T_f$, then stop. Else, go to Step 2.

In Section 4.4, we give a description of the stopping criterion and cooling scheme used for our computational results.

Tabu search

Like SA, TS (see [32], [40]) moves from one solution to another with the new solution being possibly worse than the one before. However, TS systematically searches the neighborhood and selects the best solution found, even if this solution is worse than the current solution. During the process, a tabu list is kept which contains solutions, or properties of the solutions, the heuristic is not allowed to accept. See Section 2.4.2 for a more detailed description of this method. Summarizing, our implementation of TS is as follows, where $\bar{Q}$ denotes the current best solution:

Step 1. Generate an initial solution Q using some heuristic and determine its objective function value, $f(Q)$. Set $\bar{Q} = Q$.

Step 2. Select the neighbor solution Q' of solution Q that has the lowest objective function value and that is not tabu. Solution Q', or a property of solution Q', enters the tabu list and the oldest entry is deleted. Set $Q = Q'$.

Step 3. If $f(Q') \leq f(\bar{Q})$, then set $\bar{Q} = Q'$.

Step 4. If the stopping criterion is met, then stop. Else, go to Step 2.

In Section 4.4, we define the length of the tabu list and give a description of the used solution property.

4.3.4 Shifting bottleneck heuristics

Besides pure constructive or improvement heuristics, also some combined approaches exist. The SBH [2] is such a combination. In each step of the procedure, one OR is selected for which the sequence of surgeries is determined. After this, a subset of all already scheduled ORs are rescheduled. This can be seen as an

improvement step. In the following paragraphs, these steps of SBH are further explored.

The first step in our SBH is determining the order in which the ORs are selected. Following the principle of SBH, we aim to choose the order for which the best resulting solution is achieved. We consider three options, namely (1) select the ORs in order of increasing number of surgeries, (2) select the ORs in order of decreasing maximum surgery duration, and (3) select the ORs in order of decreasing average surgery duration. The first option selects the OR that creates the smallest number of BIMs, and therefore, contributes the least to creating small BIIs. An advantage of this option is that an OR with a small number of surgeries can be rescheduled faster than an OR with a large number of surgeries. This can reduce the overall run time, because ORs that are added early in the process are rescheduled more often. The second option is considered because long surgeries can result in large BIIs. However, following the same reasoning we can choose the third option. Preliminary results have shown that, in most cases, the third option results in a shorter runtime and smaller BIIs than the first and second option.

The next step is to determine a schedule for the selected OR. As in practical instances there are only a few surgeries scheduled in each OR, we have chosen to enumerate all possible sequences of the surgeries. For each of these sequences, we determine the three largest BIIs with respect to the BIMs of already scheduled ORs. We choose the sequence that creates the lexicographic smallest three largest BIIs.

The final step is to reschedule some already scheduled ORs to reduce the three largest BIIs. Note that the maximum BII can not only be reduced by resequencing the surgeries of the OR which contains this maximum BII, but also by resequencing the surgeries of other ORs. This resequencing can split up the maximum BII by scheduling a BIM during this BII. We choose to reschedule the OR for which the completion time of one of the preassigned surgeries creates the start or end time of the smallest BII. The reasoning for this choice is that we expect to reduce the maximum BII by increasing the smallest BII. The selected OR is rescheduled by using the same technique as in the previous step. Note that this rescheduling method can also be used as an improvement heuristic for the constructive heuristics.

4.4 Computational results

We have tested the heuristics described in Section 4.3 on several instances. These instances are created based on information derived from ten years of empirical data of the OR department for inpatients in the Erasmus Medical Center. The number of ORs varies from 2 to 16 and all ORs are assumed to be generic as opposed to be dedicated to a specialty, i.e., each emergency surgery can be scheduled in all ORs. Each instance consists of data for 100 periods of five days and all ORs start at the same time and are opened eight hours a day. The solution methods are implemented in AIMMS 3.10 and run on an Intel Core2 Duo CPU P8600 2.40 GHz with 3.45 GB RAM. The parameter settings for SA and TS are given in Section 4.4.1. In Section 4.4.2, we discuss the obtained results.

4.4.1 Parameter settings

To use the developed heuristics in practice, the runtime should be low. Therefore, we limit the number of iterations for both SA and TS. The parameter settings of SA and TS, which implicitly define the number of iterations, are given in the following sections.

Simulated annealing

For SA, we choose as initial solution the solution given by the 'flexible goal values' (FLEX) algorithm because FLEX is the best constructive heuristic (see Section 4.4.2). The initial temperature is chosen such that a worse solution compared to FLEX is accepted with at least probability 0.5 (see Busetti [15]). We establish this by basing the initial temperature on the upper bound Λ of the objective function value given by:

$$\Lambda = \min_{j \in J} \max_{i \in I_j} p_i. \tag{4.14}$$

This upper bound occurs when the largest surgeries of each OR are scheduled simultaneously, see Figure 4.6.

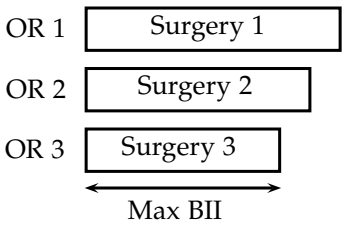


Figure 4.6: Upper bound on maximum BII

This leads to the initial temperature which is given by:

$$T_0 = \frac{f(\text{FLEX}) - \Lambda}{\ln 0.5}. \tag{4.15}$$

We set the length of the Markov chain equal to half the size of the neighborhood structure. On average we have five surgeries in each OR, and therefore, we set the length of the Markov chain to $\frac{1}{2}N(\binom{5}{2})$ which represents half the number of possible 2-exchanges. We set the reduction factor α to 0.85.

As stopping criterion, we use a threshold for the temperature. We define this threshold such that the probability of accepting a neighbor solution Q' with $f(Q') \geq f(Q) + 20$, which is a regularly occurring change in the objective value, is less than 0.01. Therefore, the final temperature is given by:

$$T_f = \frac{-20}{\ln 0.01}. \tag{4.16}$$

This cooling scheme leads to approximately 600, 1000, and 1800 iterations for 6, 10, and 16 ORs respectively.

Tabu search

As initial solution for TS, we also choose the solution given by FLEX and we set the length of the tabu list to 12. We also have to determine what property of the solution is included in the tabu list. A feasible solution is uniquely defined by the start times of each surgery. However, if we would choose this option it takes a lot of time to determine whether a solution is tabu or not. Therefore, the tabu list should not contain the solution, but some property of the solution. One candidate property is the objective function value of the solution. However, there exist many different solutions with the same objective value, and therefore, no valuable information is obtained by using this property. Another candidate property is the place of the two exchanged surgeries in the local sequence. However, the start times of the two exchanged surgeries provide much more information because they implicitly provide some information about the sequence of the other surgeries in the considered OR. A drawback is that no information is available about the sequence in the other ORs, but providing this information will require much more memory. By using the start times of the exchanged surgeries after they are exchanged, we cannot completely ensure that we do not come back to an already visited solution. However, the probability that this will happen is quite small.

When a new best solution is found, we reset the tabu list. This helps TS to fully search a new neighborhood. The process stops when after 12 iterations no new best solution is found. This stopping criterion and the length of the tabu list lead to approximately the same number of iterations for SA and TS.

4.4.2 Results for the instances

The schedules created by our solution methods are based on the expected duration of the elective surgeries. For each instance and solution method, we determine the average runtime and the average maximum BII over the 500 days. We also display the 95% confidence interval of the maximum BII. We evaluate the quality of the heuristic solution methods by the number of times the objective value equals the lower bound for the maximum BII (# LB), which is given by $\max(\lambda, \min_{i \in I} p_i)$. In addition, we display the average maximum BII and confidence intervals of the original schedules that were used in the Erasmus Medical Center to see if the proposed solution methods find a smaller maximum BII. To determine the effect of SA and TS, we also randomly generated approximately the same number of solutions as generated by SA and TS. From these solutions, we choose the solution which minimizes the maximum BII. Tables 4.1-4.3 show the results for 6, 10, and 16 ORs.

The results show that all heuristic solution methods, except SPT, provide a better solution than the original schedule used at the Erasmus Medical Center. The

	Avg. runtime (s)	Avg. max BII (min)	Conf. int. (min)	# LB
Original schedule	—	66.2	[42.2, 117.8]	2
Lower bound	—	29.4	[21.2, 46.3]	—
SPT	0.0	58.2	[40.0, 94.1]	7
Fixed goal values	0.0	46.8	[29.8, 78.4]	48
Flexible goal values	0.0	45.2	[27.8, 71.6]	72
Forward-backward	0.0	48.3	[28.4, 85.2]	35
Random	—	38.9	[28.8, 50.5]	137
SA	0.2	38.7	[27.5, 50.7]	128
TS	0.2	36.6	[23.4, 50.7]	159
SBH	0.3	38.7	[25.3, 53.7]	134

Table 4.1: Instance with 6 ORs

average length of the maximum BII decreased with more than 20%. The best heuristic method is TS for which the maximum BII decreased with more than 40%. The confidence intervals do not lead to a different conclusion. Although its average runtime is longer than the runtime of the constructive heuristics, it is still very fast (less than one second). The gap between the lower bound and the objective value of TS decreases when the number of ORs increases. This is expected because it is easier to generate BIIs close to the lower bound λ, when the number of surgeries increases. One could argue that minimizing the maximum BII would become irrelevant when the number of ORs is large or when one OR performs relatively short surgeries. However, even if surgeries have a duration of 30 minutes, it is still preferred to reduce the maximum BII to, for example, 15 minutes. And when the maximum BII is not considered while scheduling the surgeries, the resulting schedule may have a maximum BII equal to the upper bound as given in equation (4.14).

By using the ILP formulation including the lower bound as given above, we solved the instances with two and three ORs. For the instance with two ORs, we solved 499 of the 500 days to optimality. For these 499 days, the average computing time was 22 seconds, and after one hour, the integrality gap of the remaining day was 30%. The optimal solutions are equal to the lower bound for only six of the 500 days, because it is hard to reach this lower bound with only a few ORs. The solutions generated by SA, TS, and the random procedure are equal to the optimal solutions in 374, 324, and 377 of the 500 days, respectively. For the instance with three ORs, we solved 485 of the 500 days to optimality. The average computing time for these 485 days was 140 seconds, and after one hour, the average integrality gap of the remaining 15 days was 24%. The optimal solutions are equal to the lower bound for 40 of the 500 days, thus we see an increase when compared to the instance with two ORs. The solutions generated by SA, TS, and the

	Avg. runtime (s)	Avg. max BII (min)	Conf. int. (min)	# LB
Original schedule	—	48.1	[29.3, 76.5]	9
Lower bound	—	25.6	[21.2, 40.4]	—
SPT	0.0	46.8	[33.1, 70.1]	12
Fixed goal values	0.0	35.0	[22.3, 54.2]	91
Flexible goal values	0.0	33.2	[21.2, 48.4]	121
Forward-backward	0.0	37.5	[21.2, 56.6]	63
Random	—	29.0	[21.5, 40.4]	126
SA	0.3	28.9	[21.2, 40.4]	154
TS	0.3	26.8	[21.2, 40.4]	327
SBH	0.8	28.5	[21.2, 40.4]	229

Table 4.2: Instance with 10 ORs

random procedure are equal to the optimal solutions in 187, 193, and 172 of the 500 days, respectively. These results show that SA and TS already start to outperform the random procedure. For the instances with 6, 10, and 16 ORs, we interrupted the solver after 100 seconds. Because almost none of the 500 days can be solved within an hour, interrupting after one hour would lead to a total computing time of approximately 20 days per instance. Approximately 36%, 20%, and 8% of the days for the instances with 6, 10, and 16 ORs, respectively, can be solved to optimality within these 100 seconds. In addition, the results show that, for almost all of the 500 days per instance, TS gives the same or even a better solution than the solution of the truncated run of the ILP solver. The integrality gap for the days not solved to optimality is approximately 35%.

	Avg. runtime (s)	Avg. max BII (min)	Conf. int. (min)	# LB
Original schedule	—	39.1	[24.0, 54.9]	18
Lower bound	—	22.7	[21.2, 40.0]	—
SPT	0.1	41.6	[24.9, 59.4]	14
Fixed goal values	0.0	26.3	[21.2, 40.0]	200
Flexible goal values	0.0	25.4	[21.2, 40.0]	230
Forward-backward	0.1	31.5	[21.2, 50.4]	159
Random	—	22.8	[21.2, 40.0]	466
SA	0.7	22.8	[21.2, 40.0]	449
TS	0.6	22.7	[21.2, 40.0]	490
SBH	1.0	22.9	[21.2, 40.0]	468

Table 4.3: Instance with 16 ORs

When we compare TS to the random solution, we note that there is only a small difference between the average maximum BIIs. For the confidence intervals, there is only a small difference for the instance with six ORs. The difference between the average maximum BIIs is even smaller for SA. However, TS reaches the lower bound more often than the random procedure. In addition, both SA and TS perform better than the random procedure for all of the instances. We expect that the random procedure will perform worse when the number of surgeries in each OR increases or when constraints on the sequence of surgeries are included. When the number of surgeries in each OR increases, the solution space increases, and thus, the probability of selecting a good solution decreases. When constraints on the sequence of surgeries are included, the probability of selecting an infeasible solution increases, and thus, the probability of selecting a good feasible solution also decreases.

The 'flexible goal values' algorithm is the best constructive heuristic. We might expect that the 'forward-backward' algorithm would perform better, but by scheduling forwards and backwards, the algorithm leaves a gap in the middle which results in a large maximum BII. SPT is the worst constructive heuristic. This is what we expect, because SPT does not explicitly consider the objective of the BIM problem.

4.5 Simulation results

In practice, emergency patients arriving throughout the day and a change in the duration of elective surgeries may disrupt the initial schedule. With the use of simulation, we want to quantify the change in the objective function value of the created schedules due to these arrivals and the surgery duration variability. We model the arrival of emergency surgeries as a Poisson process, i.e., the interarrival times between emergency surgeries are independent and exponentially distributed. Furthermore, the duration of the emergency and elective surgeries are chosen from a lognormal distribution. The parameters of the distributions are calculated based on the data of the Erasmus Medical Center, which shows that on average four emergency patients arrive per day.

The emergency surgeries are scheduled first-come-first-served at the first available BIM after their arrival. All subsequent elective surgeries are postponed, which may result in overtime. Elective surgeries cannot start earlier than their scheduled time, because patients may not be ready for surgery.

For each evaluated schedule, the number of simulation runs is set to 200. With this value, we obtain a minimal confidence interval of 95%. We do not only determine the new average maximum BII and the 95% confidence interval denoted by 'Sim. avg. max BII' and 'Sim. conf. int. BII', but also the average waiting time of the emergency surgeries. In addition, we measure the percentage of emergency surgeries for which the waiting time is longer than 60 minutes and 30 minutes. The objective function values of the solution methods discussed in the previous section are shown in column 'Exp. avg. max BII'. 'Original schedule' is abbrevi-

	Exp. avg. max BII (min)	Sim. avg. max BII (min)	Sim. conf. int. BII (min)	Avg. wait time (min)	Perc. > 60 min	Perc. > 30 min
OS	66.2	69.2	[52.6, 95.1]	24.4	2.7	13.7
SPT	58.2	63.2	[50.6, 82.3]	23.2	2.3	12.2
Fix. GV	46.8	62.2	[49.9, 80.2]	21.1	1.4	10.5
Flex. GV	45.2	62.0	[49.8, 79.9]	21.1	1.5	10.7
F-B	48.3	62.7	[49.7, 81.2]	21.6	1.6	10.9
Random	38.9	63.4	[50.1, 80.8]	21.7	1.6	11.1
SA	38.7	63.0	[50.9, 81.0]	21.5	1.5	10.8
TS	36.6	62.5	[50.0, 82.5]	21.1	1.4	10.6
SBH	38.7	62.9	[50.0, 80.4]	21.5	1.5	10.8

Table 4.4: Instance with 6 ORs

ated to OS, 'fixed goal values' to 'Fix. GV', 'flexible goal values' to 'Flex. GV', and 'forward-backward' to 'F-B'. Tables 4.4-4.6 show the results for 6, 10, and 16 ORs.

The results show that after simulation, all heuristic methods, except SPT, provide a better solution than the original schedule in terms of maximum BII and waiting time. Therefore, it is beneficial to consider the maximum BII criteria when creating a schedule. However, the simulated average maximum BII is much larger than the expected average maximum BII as a result of the disturbances. Therefore, replanning may be needed if the disturbances of the initial schedule get too large.

	Exp. avg. max BII (min)	Sim. avg. max BII (min)	Sim. conf. int. BII (min)	Avg. wait time (min)	Perc. > 60 min	Perc. > 30 min
OS	48.1	49.3	[39.4, 62.7]	11.8	0.2	3.5
SPT	46.8	48.3	[39.2, 59.5]	12.9	0.5	4.1
Fix. GV	35.0	44.1	[37.0, 53.1]	9.8	0.1	1.9
Flex. GV	33.2	44.1	[36.7, 53.8]	10.0	0.1	2.0
F-B	37.5	44.1	[37.2, 54.4]	10.4	0.1	2.3
Random	29.0	45.1	[37.5, 54.6]	10.3	0.1	2.2
SA	28.9	44.9	[37.2, 55.3]	10.1	0.1	2.1
TS	26.8	44.3	[37.0, 53.8]	10.0	0.1	2.0
SBH	28.5	44.6	[37.4, 53.9]	10.2	0.1	2.2

Table 4.5: Instance with 10 ORs

The results also show that the heuristics that focus on spreading the BIMs as evenly as possible, i.e., 'fixed goal values' and 'flexible goal values', perform the best in the simulation study. The heuristics that focus on minimizing the three

largest BIIs, i.e., SA, TS, and SBH, perform slightly worse, because these heuristics can result in a relatively high variance of the length of the other BIIs. The heuristics that do not explicitly focus on the objective, i.e., SPT and the random procedure, perform the worst. This is a result of the better structure of the solutions generated by, for example, 'fixed goal values' and TS. This means that these solutions are more robust, which may also help when rescheduling is performed during the day.

	Exp. avg. max BII (min)	Sim. avg. max BII (min)	Sim. conf. int. BII (min)	Avg. wait time (min)	Perc. > 60 min	Perc. > 30 min
OS	39.1	36.4	[30.8, 44.4]	5.5	0.0	0.7
SPT	41.6	39.3	[32.2, 48.9]	7.6	0.1	1.6
Fix. GV	26.3	31.8	[27.3, 37.1]	4.1	0.0	0.2
Flex. GV	25.4	32.0	[27.9, 37.3]	4.2	0.0	0.2
F-B	31.5	32.1	[27.3, 37.6]	4.8	0.0	0.3
Random	22.8	32.9	[28.4, 38.4]	4.5	0.0	0.3
SA	22.8	32.6	[28.0, 38.2]	4.4	0.0	0.3
TS	22.7	32.1	[28.0, 37.3]	4.2	0.0	0.2
SBH	22.9	32.6	[28.3, 37.8]	4.5	0.0	0.3

Table 4.6: Instance with 16 ORs

We conclude that the 'fixed goal values' algorithm performs the best, because both the average waiting time and average maximum BII are the lowest. The second best heuristic is the 'flexible goal values' algorithm, followed by TS.

The simulation results also show that the average maximum BII and average waiting time decrease when the number of ORs increases. As stated in the previous section, this can be explained by a decrease in the lower bound λ. The difference between the expected and simulated average maximum BII also decreases with the number of ORs.

4.6 Conclusions and recommendations

We have introduced the BIM problem that deals with sequencing elective surgeries, which are preassigned to an OR, to reduce emergency surgery waiting time. As far as we know, this is a new type of scheduling problem which has not yet been discussed in literature. We developed and tested several heuristic solutions methods to solve this new problem.

The results of the computational study show that the expected maximum BII can be reduced by more than 20%. The simulation results show that it is beneficial to take the maximum BII into account when creating a schedule, because our heuristics outperform the original schedule when compared to the expected and simulated average maximum BII. The best expected solution is provided by

TS. However, the 'fixed goal values' algorithm provides the best simulated solution which reduces the simulated average maximum BII by approximately 10%. Therefore, we advise hospitals to use the 'fixed goal values' algorithm because it can easily be implemented and provides good solutions. The reason that the 'fixed goal values' algorithm provides a better solution than TS is that TS only focuses on minimizing the three largest BIIs.

The 'fixed goal values' algorithm can also be used to spread the workload of the holding and recovery department. This is the department in which patients are prepared before surgery and recover after surgery. When the completion times, and therefore, also the start times of surgeries are spread as evenly as possible over the day, the arrivals in and departures from the holding and recovery department are also spread over the day.

Although it may improve the results a lot, we do not consider on-line rescheduling, because in practice, it is often, for example, not possible to exchange two elective surgeries during the day. Even scheduling an elective surgery earlier in the day may not be allowed, because patients may not have arrived at the hospital yet. This means that elective surgeries can only be postponed or canceled. Another option to reduce this negative effect of disturbances is to insert breaks into the schedule, see Chapter 5.

Further research should focus on implementing practical constraints. So far, we have not considered the availability of scarce resources. In addition, the sequence of surgeries might be restricted because, for example, children should be scheduled early in the day. Adding these restrictions may limit the solution space, and therefore, also may increase the maximum BII. Furthermore, we assumed that emergency surgeries can be performed in any of the available ORs and should always be performed as soon as possible. However, in practice, some of the emergency surgeries can only be performed in a subset of the ORs which have specialized resources and some surgeries are more urgent than others.

Further research is also needed to investigate whether inserting breaks between surgeries in the schedule can increase the stability of the schedule. The simplest way to deal with disturbances caused by uncertain surgery durations is to insert a break in the middle of the day. In this way, surgeries scheduled after the midday break can mostly start on time, even if surgeries earlier that day have taken longer than expected. Furthermore, during a break, emergency surgeries can start immediately. Therefore, it is preferable to spread the breaks of the different ORs a bit over the day, such that the overlap in breaks of ORs is small. Another approach can be to plan small breaks at the end of a surgery which duration has a high variance. Additional modeling of the BIM problem and simulation experiments should clarify the effect of BIM optimization when these considerations are taken into account.

Chapter 5

Decision support system for the operating room rescheduling problem

5.1 Introduction

In this chapter, we focus on the rescheduling of surgeries, or more precisely, on the rescheduling of surgeries throughout the day. On the one hand, emergency patients who need surgery arrive throughout the day. In many hospitals, these surgeries are scheduled in one of the elective operating rooms (ORs) which disrupts the OR-schedule. On the other hand, a change in the surgery duration of elective surgeries may also disrupt the OR-schedule. Therefore, the initial planned OR-schedule may have to be adjusted throughout the day to ensure that it is still possible to execute the schedule. The new OR-schedule must fulfill quite a number of restrictions, and in addition, there are several stakeholders whose preferences and priorities must be met. Since it is hard for an OR manager to consider all these restrictions and preferences simultaneously, we develop a decision support system (DSS) which supports the OR manager with rescheduling the ORs. The approach proposed in this research is general, but the realization and used preferences are based on data from a hospital in the Netherlands.

Most existing literature focuses on operational off-line scheduling, which is done one or several days before surgery, instead of operational on-line scheduling, which is done on the day of surgery. The papers concerning operational off-line scheduling mainly focus on two methods. The first method is reserving time for emergency surgeries to minimize overtime and maximize OR utilization (see e.g. [13], [45], [59]). The second method is sequencing the elective surgeries such that the overtime caused by surgeries with a longer duration than expected is minimized (see e.g. [27], [62], [66]). In addition, some papers concerning operational off-line scheduling include the preferences of stakeholders. For example, Marcon and Dexter [67] consider the preferences of the holding and recovery departments and Marjamaa et al. [70] consider the preferences of the anesthetists.

One of the papers concerning operational on-line scheduling is the paper by Dexter [28] who examined whether moving the last surgery of the day to another OR could decrease overtime labor costs. The developed statistical strategy was based on historical data. However, in practice, often one surgeon operates in an OR on a day or part of the day, and therefore, it may not be allowed to move a surgery to another OR.

Dexter et al. [31] introduce four ordered priorities on which an OR management decision for changing the OR-schedule can be based. The first and most important priority is patient's safety. The second priority states that a surgery can only be canceled if the patient safety is not endangered. The third priority is to minimize overtime and the fourth and last priority is to reduce patient waiting times. These priorities, however, put minimizing overtime above the patients' timing preferences. For patients it is not preferred to schedule their surgery earlier and certainly not later in the day. In addition, no priority considers the workload level on other departments like wards, the holding department, and the recovery department.

Another paper of Dexter et al. [30] considers the sequencing of urgent surgical cases. They propose a sequencing which is based on the following three objectives: (i) minimize the average waiting time of surgeons and patients, (ii) sequence the surgeries in order of appearance, and (iii) schedule the surgeries in order of medical urgency. However, none of these objectives consider the preferences of the elective patients and other departments.

It seems that none of the existing papers on OR rescheduling considers the preferences and priorities of all the stakeholders simultaneously. This research tries to fill this gap. Based on a given case in a Dutch hospital, we mainly focus on personal preferences and less on economical preferences, because patient and personnel satisfaction is highly important in the Netherlands due to shortage of personnel and competition between hospitals. In Section 5.2, we discuss the stakeholders and their restrictions and preferences which are based on a survey performed at the Isala Clinics, which is the above mentioned hospital in the Netherlands. Although these restrictions and preferences may differ between hospitals, the principal ideas of the method developed in this chapter should be applicable for other hospitals too. Because we want to develop a DSS that generates adjusted OR-schedules within a short amount of time, we first analyze the problem to determine which changes are preferred based on the preferences of the stakeholders. Therefore, we introduce an Integer Linear Program (ILP) in Section 5.2 which incorporates the restrictions of the stakeholders and has as goal to minimize the deviation from the preferences of the stakeholders. Although we prove that the problem is strongly $\mathcal{NP}$-hard for two or more ORs, we are able to solve this ILP due to the moderate size of the instance. The solutions to the ILP are discussed in Section 5.3 and used to determine the changes made in an optimal OR-schedule for instances of the Isala Clinics. These changes are incorporated as decision rules in the DSS which is described in Section 5.4. As we incorporate the preferences of all stakeholders and as these preferences are given by the stakeholders themselves, we do not expect that the resulting changes are subject to psychological bias as mentioned in other papers (see e.g. [29], [63]). The developed DSS is tested by means of a simulation study to determine what improvements can be made to the OR-schedule when the developed DSS is used in practice. The computational results of this simulation study are given in Section 5.5. Section 5.6 draws conclusions and gives recommendations for further research.

5.2 Problem formulation

In this section, we give an introduction to the OR rescheduling problem and we introduce an ILP model which can be used to determine a new OR-schedule throughout the day. The ILP includes all relevant constraints that are imposed on the OR-schedule; e.g., the availability of a patient, as well as the availability of an OR with OR assistants, a surgeon, and an anesthetist. In addition, the capacity of the holding and recovery department are considered. A detailed description of the constraints is given in the following sections.

The objective of the ILP is to minimize the deviation of the preferences for the involved stakeholders. When the OR-schedule deviates from these preferences some penalty costs are incurred and the weighted sum of these penalty costs is minimized. There can be, for example, penalty costs for deviating from the scheduled start time of a surgery or for the amount of resulting overtime. In addition, we minimize the number of canceled patients, as this is not preferred by any of the stakeholders. The developed ILP can also be used to determine whether a proposed OR-schedule is feasible or not, and in case it is feasible, to calculate the deviation from the preferences of the stakeholders.

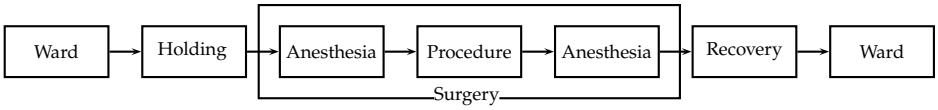


Figure 5.1: Patient process

Before we introduce the model, we first give a short description of the process a patient follows on the day of surgery in the Isala Clinics (see Figure 5.1). On or before the day of surgery, the patient is admitted on a ward where he/she is prepared for surgery. Some time before surgery, the patient is transported to the holding department where the patient is further prepared for surgery. Then, the patient is transported to the operating room where the anesthetist administers anesthesia. After this, the surgeon performs the surgical procedure. When the surgical procedure is finished, the anesthetist reverses the anesthesia, and then, the patient is transported to the recovery department where he/she recovers from the effects of the anesthesia. At the time these effects have completely worn off and the patient's condition is considered stable, he/she is transported back to the ward. Emergency patients are not first admitted on one of the wards, but are directly transported to the holding department or, in some very urgent cases, to the operating room in which the surgery will take place. After surgery, these emergency patients follow the same path as the elective patients.

For the modeling, we discretize an OR-day into T time periods which have a length of δ minutes. The length of one OR-day is therefore δT minutes. We denote by time $t \in T$ the period $((t-1)\delta, t\delta]$. The set of ORs is given by set J and consists of M ORs. The start time of OR $j \in J$ is denoted by s_j and the end time by f_j. The set of surgeries is given by set I and consists of N surgeries. The subset $I_j \subseteq I$ denotes

the surgeries that are scheduled in OR $j \in J$ and $o_i \in J$ denotes the assigned OR for surgery $i \in I$.

The initial OR-schedule, which is given at the beginning of the day and which has already been determined one or several days before, is defined by the assignment of the elective surgeries to an OR and the initially planned start times p_i of the elective surgeries. Each surgery has an expected duration e_i which includes the time for administering and reversing anesthesia, however, in practice, the duration of a surgery generally deviates from this duration and takes longer or shorter than expected. When a surgery takes less time than expected, and the next surgery starts at its assigned time p_i, the initial OR-schedule is not disrupted. However, it may be beneficial for the OR and other departments to schedule this next surgery earlier. When a surgery takes longer than expected, the next surgery may have to start later. This results in a shift of the not yet started surgeries in this OR. Because of this, some resource constraints may be violated. In addition to these deviations of the durations of the elective surgeries, emergency surgeries may arrive which also disrupt the initial OR-schedule. Therefore, throughout the day, a new OR-schedule may have to be created for all not started elective and emergency surgeries. In the following, we denote by set I the set of all these elective and emergency surgeries. The rescheduling is done by assigning a new start time to each surgery $i \in I$. Formally, this is expressed by binary variables S_{it}, which are one when surgery $i \in I$ starts at time $t \in T$ and zero otherwise. It is important to note that we do not allow the elective surgeries to be assigned to another OR, because each surgery has to be performed by the surgeon operating in the OR assigned to the surgery in the initial OR-schedule. Thus, all elective surgeries have to be scheduled in the same OR as in the initial schedule. Because we only reschedule the not yet started surgeries, the start time of OR $j \in J$ for the rescheduling problem is either given by the start time of the OR in the morning or the expected end time of the last started surgery in this OR.

Within the rescheduling, it may be necessary to cancel an elective surgery, for example because of an arriving emergency surgery. The decision variable U_i denotes whether elective surgery $i \in I$ is canceled or not, i.e., the variable is one when the surgery is canceled and zero otherwise. When U_i is zero, the surgery is not canceled, and therefore, a new start time must be assigned, i.e., $\sum_{t \in T} S_{it}$ must be one in this case. When a surgery is canceled, the opposite holds, i.e., if $U_i = 1$ we must have $\sum_{t \in T} S_{it} = 0$. This is ensured by the following constraint:

$$\sum_{t \in T} S_{it} = 1 - U_i, \qquad \forall i \in I. \tag{5.1}$$

Note that we do not consider the rescheduling of a canceled surgery, because we only focus on rescheduling within the day and not from day to day.

The new start time of surgery $i \in I$ should fulfill a number of constraints. It should be greater than or equal to (i) the ready time of the patient which is given by y_i, (ii) the start time of the assigned surgeon c_i which is given by d_{c_i}, and (iii) the start time of the assigned OR o_i. The following constraint ensures this:

$$S_{it} = 0, \qquad \forall i \in I, t < \max(s_{o_i}, y_i, d_{c_i}). \tag{5.2}$$

The subset $I_{MD} \subset I$ denotes the set of surgeries that should start before a certain time because of medical reasons. This medical deadline of surgery $i \in I$ is given by l_i. Furthermore, it is not allowed to cancel these surgeries, i.e., we must have $U_i = 0$ and the surgery has to start before l_i.

$$\begin{aligned} &\sum_{t=0}^{l_i} S_{it} = 1, && \forall i \in I_{MD}, \\ &U_i = 0, && \forall i \in I_{MD}. \end{aligned} \tag{5.3}$$

The decision variables S_{it} and U_i completely determine the new OR-schedule. However, to model the other restrictions and preferences some extra variables have to be defined which are introduced at the places where they are needed.

In each OR, only one surgery can be performed at a time. To model this, we need to determine whether a surgery is ongoing at time $t \in T$. For this, we introduce the binary variables B_{it} which are one when surgery $i \in I$ is performed on time $t \in T$ and zero otherwise. A surgery is ongoing on time $t \in T$ when the start time of surgery $i \in I$ is between time t and time $t - e_i$. This is shown in Figure 5.2 and expressed by the following constraint:

$$B_{it} = \sum_{\hat{t}=t-e_i+1}^{t} S_{i\hat{t}}, \qquad \forall i \in I, t \in T. \tag{5.4}$$

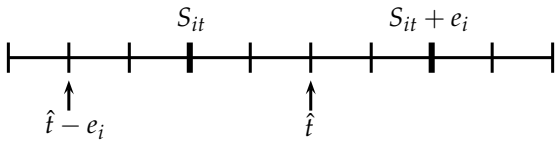


Figure 5.2: Determine B_{it}

The following constraint ensures that for each OR only one surgery can be performed at a time:

$$\sum_{i \in I_j} B_{it} \leq 1, \qquad \forall j \in J, t \in T. \tag{5.5}$$

The above constraints describe some of the hard constraints for the rescheduling process resulting from the situation in the OR. In the following section, we describe and model the involved stakeholders. For each stakeholder, we describe the tasks

the stakeholder has to perform during the day, the restrictions they impose on the OR-schedule, the impact a change in the OR-schedule has on the stakeholder and the preferences of the stakeholder. The impacts are modeled by linear constraints and penalty costs for deviating from the preferences are incorporated in the objective function. The chosen functions and corresponding parameters used to describe the preferences of the stakeholders and the penalty costs for deviating from these preferences are determined based on a survey at the Isala Clinics (see Section 5.7). Applying the model to a different hospital asks for an analysis of the preferences of the stakeholders in this hospital and may lead to other penalty cost functions. However, the principal structure of the proposed approach does not have to be adapted.

5.2.1 Stakeholders

Patient

The key stakeholder is the patient. For patients it is important that the surgery takes place at the scheduled time. Penalty costs are incurred when the new start time deviates from this preference. In order to determine the total penalty costs, we need to know the new start time of the surgery in the OR-schedule. This time is denoted by the variable W_i, and in case surgery $i \in I$ is canceled, we define W_i to be equal to the start time of surgery $i \in I$ in the initial OR-schedule.

$$W_i = \sum_{t \in T} tS_{it} + U_i p_i, \quad \forall i \in I. \tag{5.6}$$

If we now denote by Y_i the difference of the initial and new start time of surgery $i \in I$,

$$Y_i = W_i - p_i, \qquad \forall i \in I, \tag{5.7}$$

this variable Y_i is zero when surgery $i \in I$ is canceled or when the start time of surgery $i \in I$ has not changed. For emergency surgeries, we take as initial planned start time p_i the time the patient arrived at the hospital. This ensures that emergency surgeries are scheduled as soon as possible. The variable Y_i is negative when surgery $i \in I$ starts earlier in the new OR-schedule and when Y_i is positive, surgery $i \in I$ starts later. As patients judge earliness and tardiness differently, we split the variable Y_i in two cases by introducing variables Y_i^{later} and $Y_i^{earlier}$. The variable Y_i^{later} takes value Y_i when Y_i is positive and variable $Y_i^{earlier}$ takes value $-Y_i$ when Y_i is negative, which is ensured by the following constraints and the fact that the objective tries to minimize these variables:

$$\begin{aligned} Y_i^{earlier} &\geq p_i - W_i, \qquad \forall i \in I, \\ Y_i^{earlier} &\geq 0, \qquad \forall i \in I, \end{aligned} \tag{5.8}$$

$$
\begin{aligned}
Y_i^{later} &\geq W_i - p_i, && \forall i \in I, \\
Y_i^{later} &\geq 0, && \forall i \in I.
\end{aligned}
$$

Note that constraints (5.8) replace constraint (5.7) in the ILP.

Based on the survey in the hospital, we concluded that patients assign different penalty costs to different values of Y_i. To model this, a function $f_{PT}(Y_i)$, denoting the penalty costs when surgery $i \in I$ is shifted Y_i time periods, is introduced. This function is also split into two parts, namely $f_{PT}^{earlier}(Y_i^{earlier})$ and $f_{PT}^{later}(Y_i^{later})$.

Based on the patient survey, the penalty cost functions can be modeled best by step functions which are combinations of linear functions, see Figure 5.3. The specific values of the steps are also given by the questionnaire. To determine the correct value of the function $f_{PT}^{earlier}$ for a specific value of $Y_i^{earlier}$, we introduce two parameters. The first is f_k, which denotes the function value in interval $k \in K$, and the second is γ_k, which denotes the right endpoint of interval $k \in K$.

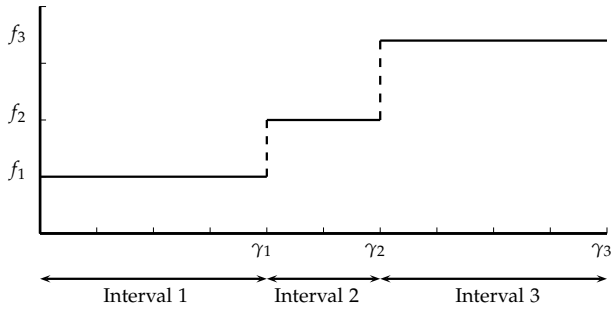


Figure 5.3: Step function

To be able to incorporate these step functions in an ILP model, we introduce binary variables Λ_{ik} which are one when $Y_i^{earlier}$ is in interval $k \in K$ and zero otherwise. This is ensured by the following two constraints:

$$
\begin{aligned}
\sum_{k \in K} \Lambda_{ik} \gamma_k &\geq Y_i^{earlier}, && \forall i \in I, \\
\sum_{k \in K} \Lambda_{ik} &= 1, && \forall i \in I.
\end{aligned}
\tag{5.9}
$$

The value of the penalty function is now determined by:

$$
f_{PT}^{earlier}(Y_i^{earlier}) = \sum_{k \in K} \Lambda_{ik} f_k, \quad \forall i \in I. \tag{5.10}
$$

The total penalty costs of the patient group are given by P_{PT} and are defined as the sum of the penalty costs of each patient:

$$P_{PT} = \sum_{i \in I} \left(f_{PT}^{earlier}(Y_i^{earlier}) + f_{PT}^{later}(Y_i^{later}) \right). \tag{5.11}$$

Note that the two penalty functions $f_{PT}^{earlier}$ and f_{PT}^{later} in general have different parameters. Because we minimize the total penalty costs, this method is only applicable for non-decreasing step functions.

Ward

Prior to surgery, the patient is admitted to a ward. On this ward, the patient is prepared for surgery. The survey showed that when a surgery starts earlier than scheduled, the workload on the ward increases if the patient is not ready yet. When a surgery starts later than scheduled, the workload can also increase. Therefore, penalty costs are incurred when there is a change in the start time of a surgery. The total penalty costs are calculated in the same way as for the patient. Based on the outcome of the survey a step function $f_W(Y_i)$ is defined, which denotes the penalty costs for the wards if the start time of a surgery is shifted for Y_i time periods. Note that we do not distinguish between a surgery being scheduled earlier or later. The total penalty costs for wards are then given by $P_W = \sum_{i \in I} f_W(Y_i)$.

Holding department

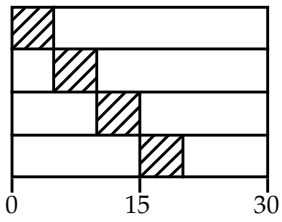
After the preparation on the ward, the patient is transported to the holding department where he/she is prepared further. The length of stay of patients on the holding department is given by v which can be longer than the preparation time needed. The holding department has a limited number of beds o_1 which provides a maximum for the number of patients treated at this department at the same time. Another limit on the number of patients who can be treated simultaneously is given by the available number of nurses at time $t \in T$ which is denoted by x_t. A nurse needs ρ minutes to prepare a patient, implying that x_t nurses can prepare at most $\frac{\delta}{\rho} x_t$ patients in time period t. Concluding, we define the capacity of the holding at time t by $\min\left(o_1, \frac{\delta}{\rho} x_t\right)$. The number of patients present on the holding on time $t \in T$ is denoted by L_t and is given by:

$$L_t = \sum_{i \in I} \sum_{\hat{t}=t+1}^{t+v} S_{i\hat{t}}, \qquad \forall t \in T. \tag{5.12}$$

This number should be smaller than or equal to the capacity of the holding which is ensured by the following constraint:

$$L_t \leq \min\left(o_1, \frac{\delta}{\rho} x_t\right), \quad \forall t \in T. \tag{5.13}$$

Note that when $\frac{\delta}{\rho} x_t \leq o_1$ for some $t \in T$, constraint (5.13) may exclude some feasible solutions (for an example, see Figure 5.4). If we want to prohibit this, we also need to schedule the preparation time of the patients. However, this increases the complexity of our problem. Note that this issue does not occur when the length of stay v equals δ which is the case for the instances used.



v = 30 minutes
ρ = 5 minutes
δ = 15 minutes
o_1 = 4 beds
x_t = 1 nurse

Figure 5.4: Excluded feasible solution

The survey at the Isala Clinics showed that the holding department prefers a leveled number of patients that are present at each point of time. Therefore, penalty costs given by the step function $f_{HD}(L_t)$, are incurred when the number of patients L_t exceeds a certain threshold. The penalty costs for different values of L_t are specified by the manager of the holding department. As at the beginning of the day (until a prespecified time ψ), personnel from the recovery department assist on the holding department (no patients are present at the recovery department at this time), penalty costs are only incurred from time ψ on. The total penalty costs are given by $P_{HD} = \sum_{t=\psi+1}^{T} f_{HD}(L_t)$.

Anesthetist

The anesthetist is responsible for administering and reversing anesthesia on one or more ORs. Therefore, he/she has to administer and reverse all anesthesias in these ORs. However, during the surgical procedure, the anesthetist does not have to be present in the OR, because the presence of an anesthesia nurse is enough. Therefore, similar to constraints (5.4) and (5.5), we include constraints which prohibits that more than one anesthesia is administered or reversed at a time in the ORs to which the anesthetist is assigned. For more details, see [49].

However, there are a few exceptions. When a surgery is complex, for example when the patient is younger than six months, the anesthetist must be present during the complete surgery which includes the surgical procedure. This means that during this time no anesthesia can be administered or reversed in one of the other assigned ORs. This is also ensured by constraints similar to constraint (5.5).

Surgeon

The surgeon is assigned to one OR and only has to perform the surgical procedure. This means that he/she does not have to be present during administering and

reversing anesthesia. The constraints that ensure this are of the same structure as constraints (5.4) and (5.5).

OR assistants

The OR assistants do not impose any restrictions on the OR-schedule. Their only preference is that overtime is minimized. Overtime can occur because of arriving emergency surgeries and surgeries whose duration is longer than expected. Therefore, penalty costs are incurred when there is overtime. The amount of overtime in OR $j \in J$ is denoted by variable O_j. The value of this variable O_j is calculated by the following constraint:

$$O_j = \sum_{i \in I_j} \sum_{t=f_j+1}^{T} B_{it}, \qquad \forall j \in J. \tag{5.14}$$

The step function $f_{OS}(O_j)$ provides the penalty costs for OR $j \in J$ when overtime of O_j time periods is incurred. The OR assistants specified the penalty costs for different values of O_j. The total penalty costs for the OR assistants are given by $P_{OS} = \sum_{j \in J} f_{OS}(O_j)$.

Recovery department

After surgery, the patient is transported to the recovery department. Here, the patient is monitored while he/she recovers from surgery. The length of stay on this department varies with the expected duration of the surgery and is given by $\max(u, \frac{1}{2}e_i)$, where u is the minimum length of stay on this department. The number of patients present at the recovery department at time $t \in T$ is denoted by Z_t and can be determined in the same way as for the holding department. The capacity of the recovery department is restricted by the number of beds o_2.

Another restriction is given by the number of patients who can be treated simultaneously, which depends on the number of available nurses r_t at time $t \in T$. Each nurse can monitor φ patients at a time, and therefore, φr_t patients can be treated simultaneously. Combining this with the number of beds o_2, the capacity of the recovery at time $t \in T$ is given by $\min(o_2, \varphi r_t)$. The number of patients present on the recovery department at time $t \in T$ should be less than or equal to this capacity. This is ensured by constraints that are of the same structure as constraints (5.12) and (5.13).

Like the holding department, the recovery department also prefers a leveled number of patients that are present at each point of time. Therefore, penalty costs are incurred when the number of patients exceeds a certain threshold. This is modeled by the step function $f_{RC}(Z_t)$ which provides the penalty costs incurred when Z_t patients are present at time $t \in T$. The total penalty costs for the recovery are then given by $P_{RC} = \sum_{t \in T} f_{RC}(Z_t)$.

Radiology department

For some surgeries, an X-ray machine is needed during surgery. These surgeries are given by the set $I_{RL} \subseteq I$. For these surgeries a radiology technician should be present during administering anesthesia and the surgical procedure. This means that he/she does not have to be present during reversing anesthesia. The set of radiology technicians is given by set K and consists of χ radiology technicians. We restrict the number of required radiology technicians D_t at time $t \in T$ to be smaller than or equal to the number of available radiology technicians. The constraints that ensure this are similar to constraints (5.4) and (5.5).

The survey showed that it is important for the radiology department that their employees at the OR department finish as early as possible such that they can carry out other work at the radiology department. Therefore, penalty costs are incurred when a radiology technician finishes later than needed, i.e., when the time the radiology technicians are present is longer than the time the radiology technicians are needed. In the following, we show how this is incorporated in the ILP.

For all radiology technicians $k \in K$, we determine when their work is finished at the OR. These finish times are denoted by τ_k and are calculated by determining the latest time period where at least k radiology technicians where needed. First, for each time period $t \in T$, we introduce binary variables $\tilde{D}_{tk}$ which are one when k or more radiology technicians are needed in time period $t \in T$ and zero otherwise. This is ensured by the following constraint:

$$\tilde{D}_{tk} \geq \frac{D_t - k + 1}{\chi}, \quad \forall t \in T, k \in K. \tag{5.15}$$

The finish time τ_k of radiology technician $k \in K$ is now given by the latest time period that $\tilde{D}_{tk}$ is equal to one, i.e.,

$$\tau_k \geq t\tilde{D}_{tk}, \qquad \forall t \in T, k \in K. \tag{5.16}$$

Using the above constraints and the fact that we minimize the working time of the radiology technicians, τ_k denotes the time the radiology technicians finish. However, these values do not equal the number of time periods they are actually present at the OR. To obtain this value, the start time and break time should be subtracted. The start time of the radiology technicians is given by $\min_{j \in J} s_j$, i.e., the start time of the OR department. For other hospitals, the start times of the radiology technicians may not be fixed. When this is the case, the start times can be determined with constraints similar to (5.15) and (5.16). In addition, all radiology technicians have a break of 45 minutes, i.e., $\frac{45}{\delta}$ time periods. The number of periods the radiology technicians are having a break is thus given by $v = \frac{45\chi}{\delta}$. Therefore, the number of time periods the radiology technicians are present at the OR is given by $\sum_{k \in K} \tau_k - \chi s_j - v$. This is an underestimation in case one or more radiology technicians finish before their break. However, we expect that this will rarely happen in practice.

The number of time periods the radiology technicians are actually working at the OR is given by $\sum_{i \in I_{RL}} (e_i - q_2)$, where q_2 is the amount of time it takes to reverse anesthesia and e_i is the duration of surgery $i \in I_{RL}$. Now, the variable X defined by

$$X = \left(\frac{\sum_{k \in K} \tau_k - \chi s_j - v}{\sum_{i \in I_{RL}} (e_i - q_2) + 1} \right) \tag{5.17}$$

denotes the inverse of the fraction of time the radiology technicians are busy. The step function $f_{RL}(X)$ denotes the penalty costs incurred for a value of X, specified by the radiology department, and gives the total penalty costs P_{RL} incurred.

Pathology department

During some surgeries, tissue is removed from a patient which needs to be examined by a pathologist. These surgeries are denoted by the set $I_{PA} \subseteq I$. After the surgical procedure, the tissue is transported from the OR to the pathology department. When tissue arrives after closing time, overtime is incurred. Let Q_i be an integer variable which denotes the amount of overtime which would be created by a single surgery $i \in I$, i.e., the number of time periods the tissue arrives late plus the examination duration w.

As after closing time only one pathologist is available at the pathology department, the available pathologist has to successively process the tissues that arrive late. In most cases this means that the pathologists has to work $\sum_{i \in I, Q_i > 0} W$ periods in overtime. However, sometimes tissue will arrive so late, that the amount of overtime equals $\max_{i \in I} Q_i$. Therefore, the amount of overtime Q_{total} is estimated as follows:

$$Q_{total} = \max \left(\max_{i \in I} Q_i, \sum_{i \in I, Q_i > 0} w \right). \tag{5.18}$$

When two sets of tissue arrive really late at approximately the same time, this number is a lower bound on the amount of overtime. However, this situation is not likely to occur in practice.

Again, a step function $f_{PA}(Q_{total})$ is used to express the penalty costs incurred for a value of Q_{total} and represents the total penalty costs P_{PA} incurred for the pathology department.

Logistic department

The logistic department is responsible for preparing materials needed during surgery. The materials are laid out in the order in which the surgeries are scheduled. When two surgeries are interchanged, the logistic assistants incurs penalty costs,

because they have to change the order in which the materials are laid out. Two surgeries $i \in I$ and $\hat{i} \in I$ can only be interchanged when they are scheduled in the same OR, i.e., when $o_i = o_{\hat{i}}$. These two surgeries are interchanged when $(p_i - p_{\hat{i}})(W_{\hat{i}} - W_i) > 0$, where p_i is the start time in the initial OR-schedule and W_i is the start time in the new OR-schedule. When this holds, we either have that both $(p_i - p_{\hat{i}})$ and $(W_{\hat{i}} - W_i)$ are positive or that both are negative. When both are positive, we have that $p_i > p_{\hat{i}}$ and $W_i < W_{\hat{i}}$. This means that in the initial OR-schedule, surgery $i \in I$ was scheduled later than surgery $\hat{i} \in I$ and that in the new OR-schedule, surgery $i \in I$ is scheduled earlier than surgery $\hat{i} \in I$. When both are negative, we have the opposite case.

We introduce binary variables $\kappa_{i\hat{i}}$ which are one when surgery $i \in I$ and $\hat{i} \in I$ are interchanged and zero otherwise. This is ensured by constraints (5.19), where T is the number of time periods per day. When $(p_i - p_{\hat{i}})(W_{\hat{i}} - W_i) > 0$, the variable $\kappa_{i\hat{i}}$ is set to one, however, when $(p_i - p_{\hat{i}})(W_{\hat{i}} - W_i) \leq 0$ the variable $\kappa_{i\hat{i}}$ could be set to either one or zero. But because we want to minimize the number of exchanged surgeries, the variable $\kappa_{i\hat{i}}$ gets the value zero.

$$(p_i - p_{\hat{i}})(W_{\hat{i}} - W_i) \leq T^2 \kappa_{i\hat{i}}, \quad \forall (i > \hat{i}) \in I, o_i = o_{\hat{i}}. \tag{5.19}$$

Let f_{LD} be the penalty cost incurred when two surgeries are interchanged, which value is specified by the logistic department. Then the total amount of penalty costs P_{LD} incurred for the logistic department are given by:

$$P_{LD} = \sum_{j \in J} \sum_{(i,\hat{i}) \in I_j, i > \hat{i}} \kappa_{i\hat{i}} f_{LD}. \tag{5.20}$$

5.2.2 Objective function

The goal of our model is to minimize the deviation from the preferences of the stakeholders. We denote the set of stakeholders by Π and we consider each stakeholder to be equally important. Since the order of magnitude of the cost functions introduced for the different stakeholders may differ, we have to introduce a weighted sum of the penalty costs P_π to compensate these differences. In this function, the priority β_π assigned to stakeholder $\pi \in \Pi$ is determined such that all stakeholders contribute approximately the same amount to the objective function value. The general concept of the weighted sum of penalties has furthermore the advantage that by varying the priorities, we can develop, for example, also a more patient centered OR-schedule.

Next to the penalty costs for deviation from the preferences of stakeholders, we also include penalty costs η for canceling a surgery. These penalty costs are set such that they contribute more than the combined total penalty costs of the stakeholders in case surgeries are canceled. This way, it is clear that canceling a surgery is not preferred, however, if needed it is possible to do it. Note that the resulting changes

in the OR-schedule are not risk-averse (see [85]), because by giving the objective 'minimizing the number of cancellations' a very high weight, the OR utilization is maximized. Summarizing, the objective function is given by:

$$\min \sum_{\pi \in \Pi} \beta_\pi P_\pi + \sum_{i \in I} \eta U_i. \tag{5.21}$$

5.2.3 Problem complexity

The problem introduced in the previous sections has been modeled as an ILP. The following theorem justifies this approach, since it shows that efficient exact approaches are unlikely to exist.

Theorem 5.1. *The OR rescheduling problem is strongly $\mathcal{NP}$-hard for two or more operating rooms.*

Proof. We prove the theorem by reducing 3-partition to the OR rescheduling problem. The 3-partition problem can be formulated as follows. Given positive integers $a_1, \ldots, a_{3t}$, and b with $\sum_{j=1}^{3t} a_j = tb$, do there exist t pairwise disjoint subsets $R_k \subset \{1, \ldots, 3t\}$ such that $\sum_{j \in R_k} a_j = b$ for $k = 1, \ldots, t$? The 3-partition problem is proven to be strongly $\mathcal{NP}$-hard (see Garey and Johnson [36]).

The reduction is based on the following transformation, where we set the priorities for the patient, ward, and the holding, recovery, radiology, pathology, and logistic department to zero. Therefore, we only aim to minimize overtime and the number of cancellations. Furthermore, we consider two ORs which have their own anesthetist and $6t - 2$ surgeries with the following processing times and ready times:

$$\begin{array}{lll} e_i = b, & y_i = 0, & \forall 1 \leq i \leq t-1, i \in I_1, I_{RL}, \\ e_i = a_{i-t+1}, & y_i = 0, & \forall t \leq i \leq 4t-1, i \in I_1, \\ e_i = b, & y_i = 2b(i-4t), & \forall 4t \leq i \leq 5t-1, i \in I_2, I_{RL}, \\ e_i = b, & y_i = b(i-(5t-1)), & \forall 5t \leq i \leq 6t-2, i \in I_2. \end{array} \tag{5.22}$$

The end and start times of the two ORs are:

$$s_j = 0, \quad f_j = (2t-1)b, \quad \forall j \in J. \tag{5.23}$$

The capacities of the holding and recovery departments are assumed to be larger than $6t - 2$, thus we do not have to consider the given constraints for these departments. Furthermore, only one radiology technician is available, and therefore, we have to consider the given constraints for the radiology department. Our goal is to create an OR-schedule with objective value less than or equal to zero.

First note that because of their ready times, the surgeries from $\{4t, 4t+1, \ldots, 6t-2\}$ in OR 2 have to be scheduled as in Figure 5.5 to achieve an objective value of zero, i.e., no overtime and cancellations may occur. In Figure 5.5, the grey blocks denote surgeries that need a radiology technician and the white blocks denote

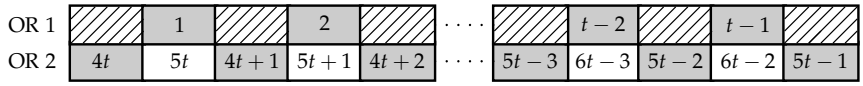


Figure 5.5: Reduction of 3-partition problem to the OR rescheduling problem

surgeries that do not need a radiology technician. Because the surgeries from $\{1, 2, \ldots t-1\}$ in OR 1 need a radiology technician, they have to be scheduled in the time intervals where the radiology technician is not busy in OR 2, i.e., as in Figure 5.5. This leaves us with t blocks of length b in OR 1 which have to be filled with the surgeries from $\{t, t+1, \ldots, 4t-1\}$ to achieve zero overtime with zero cancellations, and thus, an objective value of zero. Therefore, our problem has a solution with objective value zero if and only if there exists a solution to the 3-partition problem. □

The proof of this theorem shows that already a very restricted version of the OR rescheduling problem is strongly $\mathcal{NP}$-hard.

5.3 Computational results ILP

We tested our ILP on data from the Isala Clinics. The data consists of 1168 surgeries scheduled over 27 days. The surgeries consist of 354 emergency surgeries, 193 surgeries which need X-ray, 79 surgeries during which tissue is removed, and seven complex surgeries. The average expected duration of the surgeries is 103 minutes and the average realized duration of the surgeries is 91 minutes. Because rescheduling is only performed during working hours, we removed the emergency surgeries that start before 07:30 or after 18:00. We implemented our model in AIMMS 3.10 and solved it with CPLEX 12.1 on an AMD Ahtlon X2 Dual Core L310 1.2 GHz processor with 4 GB RAM.

In the first two sections, we discuss the parameter settings for the ILP model and the achieved results which are used to derive the decision rules for the DSS. In the last section, we determine the penalty costs for the initial OR-schedule used at the Isala Clinics and the OR-schedule realized at the end of the day. In addition, we optimize both the initial and realized OR-schedule to show what improvements potentially can be realized when the developed method is used.

5.3.1 Parameter settings

In this section, we discuss the parameter settings for the time periods and the priorities for each stakeholder. To determine the appropriate length δ of the time periods, we solved the model for time periods of 5, 10, 15, and 20 minutes. We interrupted the ILP solver after ten minutes of computation time. If after this time no optimal solution was found, we took the best solution found as our final solution. In Figure 5.6, the runtime for each combination of day and δ is given.

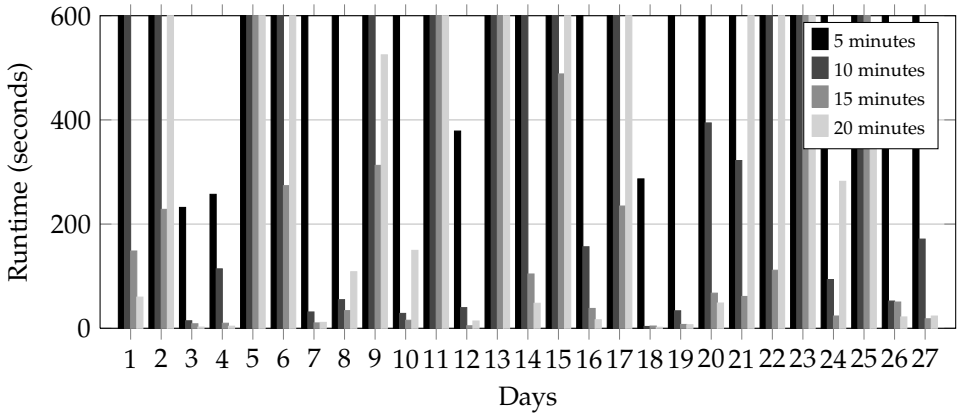


Figure 5.6: Runtime for different values of δ

We would expect that a smaller value of δ would increase the runtime of our model. For most days in our instance this holds, however, in some cases, the runtime for our model with δ equal to 15 minutes is shorter than the runtime for our model with δ equal to 20 minutes. Figure 5.7 shows that for δ equal to five minutes, an optimal solution was only found for four of the 27 days. For δ equal to 10 and 20 minutes, this number increased to 14 and 17 days, respectively. The model with δ equal to 15 minutes performs the best, because an optimal solution was found for 22 of the 27 days. This result seems to be the consequence of the input data, since most of the data is given in multiples of 15 minutes, for example, the expected surgery duration and the length of stay on the holding department.

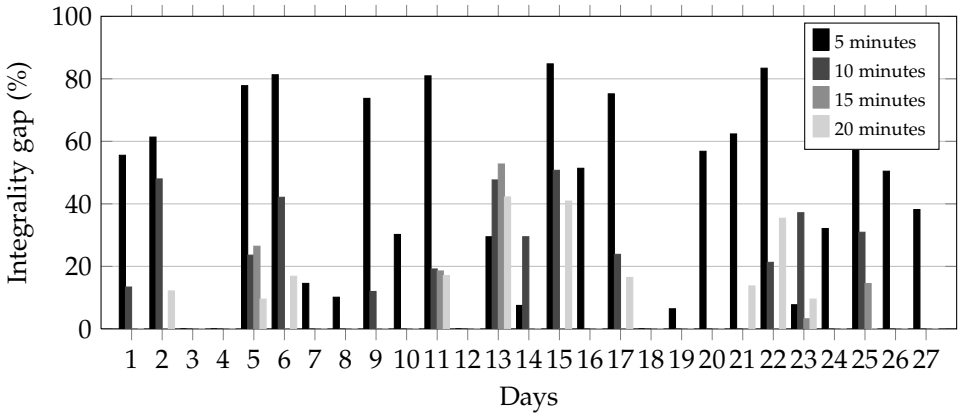


Figure 5.7: Integrality gap for different values of δ

In Figure 5.8, we give the objective function value for each combination of δ and

day. If solved to optimality, the objective value should increase when δ increases, because there is more flexibility in the OR-schedule when δ is lower, i.e., the model with $\delta = 5$, should be able to provide the same or even a better solution than the model with δ set to 10, 15, or 20 minutes. However, Figure 5.8 shows that the worst objective values are achieved when δ equals five. This is because for most days no optimal solution was found within ten minutes. From Figure 5.8, we can conclude that our model with δ set to 15 minutes results in the lowest objective function value. Combining the results for the runtime and objective function, we choose to set δ to 15 minutes for further tests.

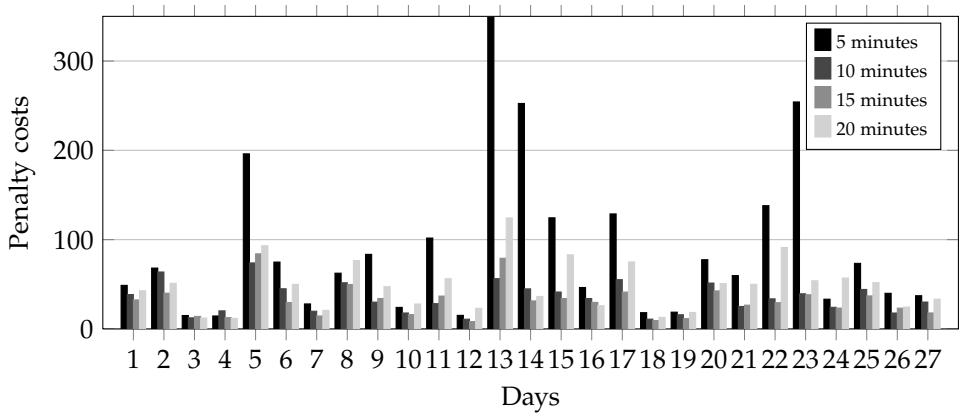


Figure 5.8: Objective function value for different values of δ

For each of the stakeholders, we have to determine its priority in the objective function. In the objective function, the total penalty costs of each stakeholder are multiplied by this priority. Our goal is that each stakeholder has approximately the same contribution in the objective function. In the following, we describe how we have determined the priority of each of the stakeholders.

	Total penalty costs	Priority	Weighted costs
Patient	7.45	0.08	0.56
Ward	5.26	0.11	0.56
Holding	0.00	1.00	0.00
OR assistants	1.11	0.50	0.56
Recovery	1.91	0.29	0.56
Radiology	1.00	0.56	0.56
Pathology	0.56	1.00	0.56
Logistics	0.91	0.62	0.56

Table 5.1: Total penalty costs and priorities

First, we solve our model where all stakeholders have priority one. Next, we adjust the priorities in such a way that the weighted cost of each of the stakeholders for the achieved solution is approximately the same. This is done by setting the priority of the stakeholder with the lowest total penalty costs to one and the priorities of the other stakeholders such that their weighted costs equal the lowest total penalty costs. Table 5.1 shows the results if this method is applied to our data, where the total penalty costs are the average total penalty costs incurred per day.

Note that the holding department does not incur any penalty costs, because penalty costs are only incurred when four patients are treated simultaneously. The constraints imposed, however, limit the number of patients present on the holding department to three. This means that the holding department is not a bottleneck in the current situation.

We conducted some further tests where we varied these priorities slightly. The results from these tests show that patients, wards, and OR assistants have opposite interests compared to the recovery, radiology, pathology, and logistic department.

5.3.2 Deriving decision rules

The main goal of applying the ILP model is to determine which adjustments to the initial OR-schedule are allowed and preferred based on the given preferences of the stakeholders. To determine this, we use our ILP model to create at three points $t \in T$ a new OR-schedule which minimizes the deviation from the preferences of the stakeholders. For each of these three scenarios, the initial OR-schedule is given as input as well as the realization of the duration of the surgeries that started before time t. These realized durations may change the initial OR-schedule, because this schedule was based on the expected durations. Since we cannot change the OR-schedule for the already started surgeries, we start rescheduling at the new start time s_j of OR j, which we define as the end time of the last started surgery before time t. In addition, we schedule not yet started emergency surgeries that arrived before time t. We assign a new start time S_{it} to each elective and emergency surgery that has not started at time t or, when allowed, cancel this surgery such that the resource constraints are fulfilled and the deviation from the preferences is minimized. The three scenarios are summarized below.

Scenario 1 After 10 a.m.: In this scenario, the realized durations of the surgeries that started before 10 a.m. are known. An emergency surgery is only included if it arrived before 10 a.m..

Scenario 2 After 12 p.m.: In this scenario, the realized durations of the surgeries that started before 12 p.m. are known. An emergency surgery is only included if it arrived before 12 p.m..

Scenario 3 After 2 p.m.: In this scenario, the realized durations of the surgeries that started before 2 p.m. are known. An emergency surgery is only included if it arrived before 2 p.m..

These three scenarios are used to determine what adjustments our model makes to the OR-schedule. For each of the scenarios, we determine how often one of the following adjustments occurred: (i) shifting a surgery, (ii) exchanging the sequence of two surgeries, and (iii) canceling a surgery. In addition, we determine how often a break of a certain length was scheduled between two surgeries. The results are shown in Table 5.2.

	10 a.m.	12 p.m.	2 p.m.
Reschedulable surgeries	566	416	213
Shifted surgeries	375	297	176
Exchanged surgeries	1	0	0
Canceled surgeries	0	0	1
No break	264	183	71
Break 15 min.	166	112	66
Break 30 min.	62	33	21
Break 45 min.	22	19	11
Break $>$ 45 min.	38	37	13
Mean break (minutes)	15.84	18.71	19.78

Table 5.2: Results scenario 1, 2, and 3

Table 5.2 shows that shifting a surgery is the most frequently used adjustment and often we see that a break is scheduled between two surgeries. The average length of a break is 15 to 20 minutes. When we only consider OR utilization, this may not seem to be optimal, however, these breaks can improve the perceived workload of other departments or may be necessary to fulfill the resource constraints. From the results we conclude that only two types of adjustments are preferred to be used. A surgery can be shifted or a break can be scheduled between two surgeries. This means that the order of surgeries stays the same during the day. So when we only allow these two adjustments, the number of feasible solutions decreases significantly, because we only have to consider one sequence of the surgeries instead of all possible sequences. Based on this, it is possible to develop a simple heuristic to determine a good OR-schedule. A further benefit of this is that we do not need an expensive ILP solver to implement our approach.

Following the above considerations, we have incorporated this simple heuristic in a DSS which is described in Section 5.4.

5.3.3 Potential improvements

To determine what improvements the DSS could potentially make compared to the OR-schedules used at the Isala Clinics, we calculated, using the ILP model, the optimal initial OR-schedule based on the expected surgery durations. Note that for this optimization the assignment of the surgeries to an OR is given as in the given

initial OR-schedule. Thus, we only change the sequences of the surgeries in each OR. None of the surgeries can be canceled, and because it is an initial OR-schedule, the change in the start time of the surgeries is not incorporated in the ILP model. This implies that the penalty costs for the patients, wards, and logistic department are zero. Also, there are no emergency surgeries to be scheduled, because they have not arrived yet. This resulting fourth scenario is summarized below.

Scenario 4 Initial OR-schedule: In this scenario, we compare the initial OR-schedule used at the Isala Clinics with a new OR-schedule determined by the ILP model, where we rescheduled all elective surgeries. Therefore, there are no emergency surgeries to be scheduled and only the expected duration of each surgery is given.

In addition, we want to determine what improvements the DSS could make to the realized OR-schedule. Filling in the actual surgery durations in the initial OR-schedule will most likely result in an infeasible solution and because we do not know the exact decisions the OR manager will make, we can only guess what improvements can be achieved. To provide an upper bound on these improvements, and thus, a lower bound on the penalty costs, we optimized the realized OR-schedule and compared it to the realized OR-schedule provided by the Isala Clinics. The realized OR-schedule of the Isala Clinics denotes how the surgeries were actually performed. This means that several changes have been made to the initial OR-schedule, and therefore, we can determine the penalty costs achieved for these changes. For the optimal realized OR-schedule, we assume that all realized durations of the elective and emergency surgeries are considered to be known in advance and also the canceled surgeries are taken into account. In practice, this information is not known beforehand, and thus, this optimal realized OR-schedule can not be achieved in practice. However, it provides a bound, and therefore, an indication of the room for improvement.

To define the input more precisely, we take all elective surgeries with their realized duration that were planned in the initial OR-schedule, and in addition, we include all performed emergency surgeries with their realized duration. For the canceled surgeries, we use the given expected surgery durations. For this scenario, it can happen that surgeries are canceled or that their start time changes, which results in penalty costs for patients, wards, and the logistic department. This fifth scenario is summarized below.

Scenario 5 Realization: This scenario consists of all the elective surgeries scheduled in the initial OR-schedule and all emergency surgeries that arrived during the day. The realized duration of all surgeries is known.

Table 5.3 provides the average total penalty costs per day for each of the stakeholders and compares the initial and realized OR-schedule of the Isala Clinics to the optimal OR-schedules created by our ILP model. In the total costs, the priorities of the stakeholders are included.

	Initial OR-schedule		Realized OR-schedule	
Total penalty costs	Original	Optimal	Original	Optimal
Cancellation	0.00	0.00	66.67	22.22
Patient	0.00	0.00	40.91	36.50
Ward	0.00	0.00	34.40	28.56
Holding	0.00	0.00	0.00	0.00
OR assistants	0.00	0.68	6.43	4.20
Recovery	4.93	0.74	8.85	3.11
Radiology	1.63	0.56	2.41	1.07
Pathology	0.64	0.15	0.70	0.51
Logistics	0.00	0.00	9.26	5.93
Total costs	3.00	1.03	87.03	35.82

Table 5.3: Results scenario 4 and 5

Table 5.3 shows that in Scenario 4, the initial OR-schedule, the total penalty costs for the recovery and radiology department decrease. However, this can only be achieved by scheduling some surgeries in overtime. This follows from the slight increase of the total penalty costs for the OR assistants. For Scenario 5, the realization, the results show that the objective function value is reduced by approximately 60%. The major decrease is caused by the reduction of the number of cancellations. Also, the total penalty costs for the recovery department decrease significantly. In practice, the penalty costs for the realized OR-schedule will lay somewhere between 35.82 en 87.03 when the DSS, discussed in the next section, is used.

Concluding, our model can potentially improve the initial and realized OR-schedule significantly.

5.4 Decision support system

To make our method applicable in practice, we have developed a DSS which can be used by the OR manager. We incorporated the two decision rules that are derived from the results in Section 5.3.2. The first decision rule is that the order of surgeries must be maintained, but that a surgery can be shifted in time. In addition, the second decision rule states that it is allowed to schedule a break of at most one hour between two surgeries. This may help to decrease the perceived workload of several stakeholders and may be necessary to fulfill the resource constraints.

During the day, the OR-schedule must be adjusted, because of arriving emergency surgeries and elective surgeries that take shorter or longer than expected. The user can indicate for which OR the schedule should be adjusted. Because we have to consider all preferences and restrictions with respect to the other ORs, adjusting the schedule of the chosen OR is not straightforward. However, since we only have small instances, we evaluate, by means of complete enumeration, all

	Priority	Planning		Option 1		Option 2		Option 3	
Stakeholder		**Penalty costs**	**Weighted costs**	**Penalty costs**	**Weighted costs**	**Penalty costs**	**Weighted costs**	**Penalty costs**	**Weighted costs**
Patient	0.05	26.20	1.31	25.90	1.26	26.20	1.31	26.20	1.31
Ward	0.08	26.60	2.13	25.20	2.07	26.60	2.13	26.60	2.13
Holding	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
OR assistants	0.54	0.00	0.00	0.00	0.00	0.00	0.00	0.60	0.32
Recovery	0.16	13.00	2.08	11.00	1.76	13.00	2.08	13.00	2.08
Radiology	0.20	2.00	0.40	2.00	0.40	2.00	0.40	2.00	0.40
Pathology	0.10	0.70	0.07	0.70	0.07	0.70	0.07	0.70	0.07
Logistics	0.43	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total costs			**5.99**		**5.56**		**5.99**		**6.31**

Figure 5.9: Decision support system - penalty costs

possible solutions for this OR with respect to the two decision rules. This means that between each two surgeries a break is scheduled with a duration that varies between zero and four time periods. When n surgeries have to be rescheduled, this results in 5^{n-1} possible solutions. In most cases n will be smaller than five, which gives approximately 100 possible solutions. After all possible solutions are evaluated, the DSS presents the three best options to the user. Only feasible solutions with respect to the constraints described in Section 5.2 are considered. A screen-shot of the DSS is shown in Figure 5.9. The first column of the screen-shot gives the specified priorities of all stakeholders. These values can be changed to create, for example, a patient centered OR-schedule. The next column shows the penalty costs and weighted costs of the current OR-schedule. The last three columns show the three best OR-schedules with their penalty costs and weighted costs from which the user can choose. In addition, the DSS provides a Gantt chart of the current OR-schedule and the three best feasible OR-schedules (see Figure 5.10).

The DSS gives insight in how other departments are influenced by a change in the OR-schedule by denoting the penalty costs and weighted costs incurred for each stakeholder. This can convince surgeons that it can be useful to schedule a break between two surgeries. In addition, the DSS can determine whether the initial OR-schedule is feasible or not by checking all constraints given in Section 5.2. The system denotes for each constraint how many times it is violated in the proposed OR-schedule. It furthermore can be used to adjust the schedule such that it is feasible. Also, the penalty costs for the proposed OR-schedule are calculated which gives an indication of how good the schedule is. The last advantage of the DSS is that the realized OR-schedule can be evaluated. This way, the OR manager can learn from the decisions made in the past.

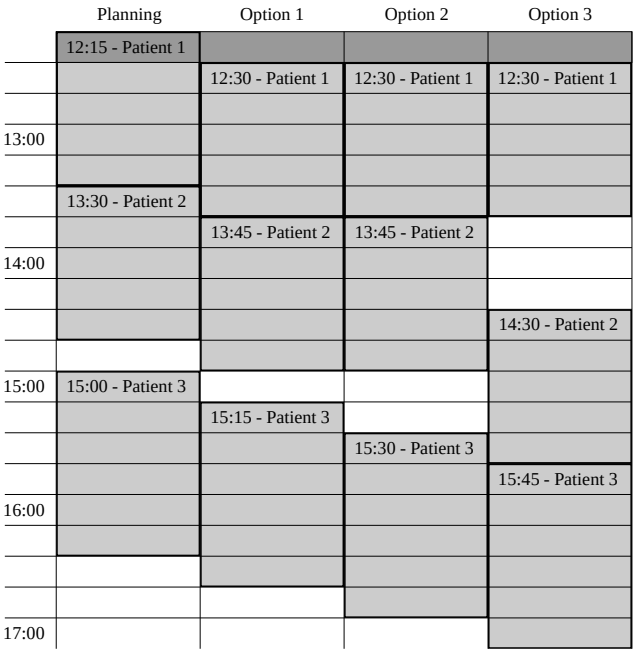


Figure 5.10: Decision support system - Gantt chart

5.5 Simulation study DSS

To determine the effect of using the DSS in practice, we tested the developed DSS on the instance given in Section 5.3 by means of a simulation study. In this study, we simulate the use of the DSS by rescheduling an OR immediately when it is disturbed. We choose to evaluate this rescheduling strategy, because we believe that this strategy will provide the best results.

At the beginning of each of the days in the considered instance, only the information that at that time is also known in practice is given. More specific, only the initial OR-schedule is known, which is defined by the assignment of the elective surgeries to an OR and the initially planned start times P_i of the elective surgeries together which their expected duration E_i. This means that the emergency surgeries are not known in advance in contrast to the considered Scenario 5.

For each point in time during the simulation, we insert the arrived emergency surgeries into the OR-schedule and we update the surgery duration for each of the ongoing surgeries. For each arrived emergency surgery, we set the planned start time equal to the time of arrival and set the duration equal to the expected surgery duration. In addition, each emergency surgery has to be performed in the same OR as it was performed in the realized OR-schedule of the Isala Clinics.

For the ongoing surgeries, we update the duration according to the following rules. When the realized duration of a surgery is less than the expected duration,

we set the duration equal to the expected duration until the surgery is finished. At the moment in time the surgery is finished, we set the duration equal to the realized duration. When the realized duration of a surgery is larger than the expected duration, we set the duration equal to the maximum of the expected duration and the already executed duration thus far. When the already executed duration equals the realized duration, the surgery is finished, and thus, the duration is set to the realized duration. In the next step of our simulation, we determine for each OR whether it becomes empty or whether an elective surgery which is scheduled to start at this point in time cannot be started. When an OR becomes empty, a new surgery may be started which could potentially improve the OR-schedule for one or more of the stakeholders. In case an elective surgery which is scheduled to start cannot be started, we have the situation that an already started surgery assigned to the same OR takes longer than its expected duration. When one of these two cases occur, we reschedule this OR to decrease the penalty costs or to make the OR-schedule feasible again. We reschedule this OR with respect to the fixed schedule in the other ORs as described in Section 5.4.

	Realized OR-schedule		
Total penalty costs	Original	Optimal	DSS
Cancellation	66.67	22.22	0.00
Patient	40.91	36.50	29.50
Ward	34.40	28.56	21.44
Holding	0.00	0.00	0.00
OR assistants	6.43	4.20	12.21
Recovery	8.85	3.11	14.70
Radiology	2.41	1.07	2.78
Pathology	0.70	0.51	1.14
Logistics	9.26	5.93	0.00
Total costs	87.03	35.82	45.13

Table 5.4: Results simulation study DSS

The results of the simulation study in Table 5.4 show that the DSS provides a much better solution than the original realized OR-schedule of the Isala Clinics. However, as expected, the penalty costs for the DSS are higher than the penalty costs for the optimal realized OR-schedule. When compared to the original realized OR-schedule, the total penalty costs are reduced by approximately 50% for the DSS and by approximately 60% for the optimal realized OR-schedule. In addition, the results show that the penalty costs for the patients and wards decrease when compared to the realized OR-schedule of the Isala Clinics and the optimal realized OR-schedule, however, the penalty costs for the OR assistants and the recovery, radiology, and pathology departments increase. The penalty costs for the mentioned stakeholders increase because surgeries cannot be canceled or exchanged,

and thus, more surgeries have to be done in overtime. Concluding, the use of the DSS provides a better trade-off between the preferences of the involved stakeholders than the original realized OR-schedule of the Isala Clinics and by this reduces the incurred penalty costs significantly.

5.6 Conclusions and recommendations

In this chapter, we considered the problem of rescheduling surgeries on the day of execution. We formulated an ILP which determines the best adjusted OR-schedule at a given point in time. The results show that patients, wards, and OR assistants have opposite interests compared to the recovery, radiology, pathology, and logistic department. Furthermore, the achieved results show that, with a few exceptions, the only used adjustments are (i) shifting surgeries, and (ii) scheduling breaks between two surgeries. These two decision rules are incorporated in a developed DSS. This system determines the best adjusted schedule for one OR with respect to the given constraints and gives insight in how the workload of stakeholders is influenced by adjusting the OR-schedule throughout the day. The simulation study shows that by using this DSS, less surgeries are canceled and patients and wards are more satisfied when compared to the original realized OR schedule, but also that the workload of several departments increases to compensate this.

A drawback of the developed DSS is that the decision rules may not be applicable when the priorities of the stakeholders change. A change in these priorities for the ILP can result in, for example, more exchanges or cancellations of surgeries. However, this is not expected in hospitals that have a similar group of stakeholders as the Isala Clinics, because these two adjustments are less preferred than shifting a surgery.

Another aspect of the DSS, which may be seen as a drawback, is that it only improves the OR-schedule for one OR at a time. The idea for such a type of approach comes from the Shifting Bottleneck Heuristic [2], where in each step the schedule of the bottleneck resource is optimized with respect to the schedule for the other resources. As for the Shifting Bottleneck procedure, our approach is only a heuristic approach, and thus, will not result in optimal solutions. However, the results of the simulation study show that the DSS performs relatively good when compared to the optimal OR-schedule. In addition, the Isala Clinics do not prefer to reschedule multiple ORs at once, because then the process of optimization may be unclear to the user and the necessary changes to the OR-schedule at one point in time may be quite large resulting in a decrease of the acceptance of the achieved results.

Further research could focus on including the Central Sterile Supply Department (CSSD) into the model. This department prepares the instrument sets needed for a surgery. When a surgery is added to the OR-schedule during the day, this may influence the workload on the CSSD. In addition, the CSSD may impose some extra constraints on the OR-schedule.

There are several ways in which the developed DSS can be used, for example,

reschedule an OR immediately when it is disturbed or reschedule all ORs at some moments in time. The last example also raises the question in what order the ORs should be rescheduled. Therefore, it would be interesting to investigate the best way to use the DSS.

5.7 Appendix

The preferences and restrictions of the stakeholders were determined by means of interviews and we conducted a survey to determine the penalty costs the stakeholders assign to changes in the OR-schedule. In this appendix, we describe for each stakeholder the survey used and the resulting penalty costs.

Patient

By means of interviews with patients and nurses at the wards, we learned that patients do not prefer a change in the start time. Because it was not possible to ask the patients what penalty costs they would assign to different changes in the start time, we asked the nurses at the wards to fill in the survey. We received a completed survey from 29 nurses. The questions of the survey are depicted in Figure 5.11 and the results in Figure 5.12.

Penalty Costs Patients

Could you please indicate below how many points patients would assign to each of the changes in the start time of the surgery?
0 points means that this change is acceptable for the patient and 5 points means that this change is unacceptable.

The surgery is performed ...	0	1	2	3	4	5
... more than 2 hours earlier.	o	o	o	o	o	o
... 1 to 2 hours earlier.	o	o	o	o	o	o
... $\frac{1}{2}$ to 1 hour earlier.	o	o	o	o	o	o
... $\frac{1}{2}$ hour later to $\frac{1}{2}$ hour earlier.	o	o	o	o	o	o
... $\frac{1}{2}$ to 1 hour later.	o	o	o	o	o	o
... 1 to 2 hours later.	o	o	o	o	o	o
... 2 to 3 hours later.	o	o	o	o	o	o
... 3 to 4 hours later.	o	o	o	o	o	o
... 4 to 5 hours later.	o	o	o	o	o	o
... more than 5 hours later.	o	o	o	o	o	o

Thank you for filling out this survey!

Figure 5.11: Survey patients' preferences

Ward

The interviews with the nurses at the wards also revealed that the nurses do not prefer a change in the start time of a surgery. Therefore, we asked the nurses to fill out the same survey as depicted in Figure 5.11, however, this time, we asked them to indicate how many points they would assign to each of the changes in the start time of the surgery. We received a completed survey from 29 nurses for which the results are depicted in Figure 5.12.

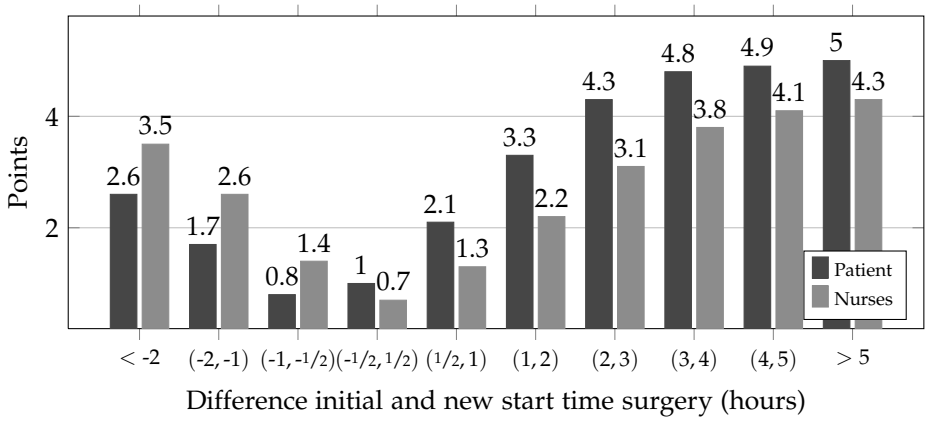


Figure 5.12: Results patients' and nurses' survey

Holding department

Interviews with nurses at the holding department revealed that they prefer a leveled number of patients at each point in time. The manager of the holding department indicated by assigning points which number of patients is preferred by the department, where zero point means that the number of patients is acceptable and five points means that the number of patients is unacceptable. The results can be found in Table 5.5.

Number of patients	Points holding	Points recovery
0	0	0
1	0	0
2	0	1
3	0	3
4	1	5

Table 5.5: Results holding and recovery department survey

OR assistants

By means of interviews with the OR assistants, we learned that OR assistants prefer a minimal amount of overtime. We asked the OR assistants to fill out a survey in which they could assign points to different amounts of overtime. The survey, depicted in Figure 5.13, was filled out by 36 OR assistants. The results of this survey are shown in Figure 5.14.

Penalty Costs Overtime

Could you please indicate below how many points you would assign to each amount of overtime? 0 points means that this amount is acceptable and 5 points means that this amount is unacceptable.

The amount of overtime is:	0	1	2	3	4	5
0 to 15 minutes	o	o	o	o	o	o
15 to 30 minutes	o	o	o	o	o	o
30 to 45 minutes	o	o	o	o	o	o
45 to 60 minutes	o	o	o	o	o	o
60 to 75 minutes	o	o	o	o	o	o
75 to 90 minutes	o	o	o	o	o	o
90 to 105 minutes	o	o	o	o	o	o
105 to 120 minutes	o	o	o	o	o	o
more than 120 minutes	o	o	o	o	o	o

Thank you for filling out this survey!

Figure 5.13: Survey OR assistants' preferences

Recovery department

Interviews with nurses at the recovery department revealed that they prefer a leveled number of patients at each point in time. The manager of the recovery department indicated by assigning points which number of patients is preferred by the department, where zero point means that the number of patients is acceptable and five points means that the number of patients is unacceptable. The results can be found in Table 5.5.

Radiology department

Interviews with radiology technicians revealed that they prefer to finish as soon as possible with their work at the OR such that they can assist at the radiology department. As measurement of deviating from this preference, we take the percentage of time radiology technicians are present but not working at the OR. The bigger this percentage, the more points are assigned. The assigned points per deviation are shown in Table 5.6.

Percentage	Points
0-10%	0
10-25%	1
25-37.5%	2
37.5-50%	3
50-60%	4
>60%	5

Table 5.6: Results radiology department

Pathology department

By means of interviews with the pathology department, we learned that the employees prefer a minimal amount of overtime. We asked the manager of the pathology department to assign points to different amounts of overtime, where zero points means that the amount of overtime is acceptable and five points means that the amount of overtime is unacceptable. The results are shown in Figure 5.14.

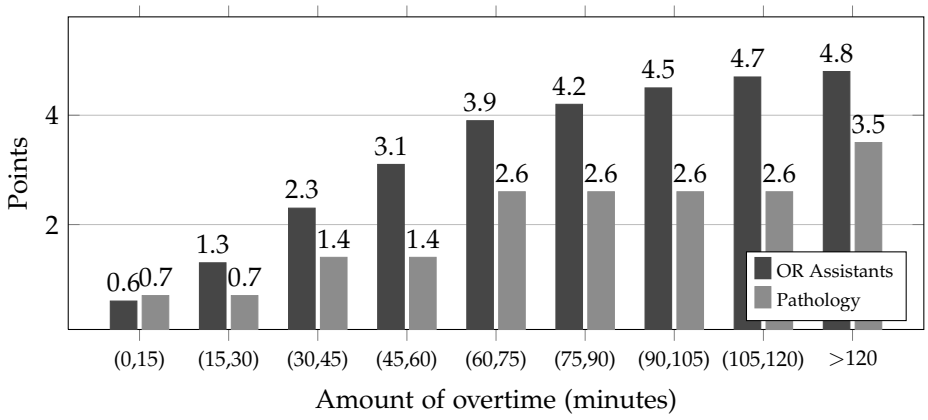


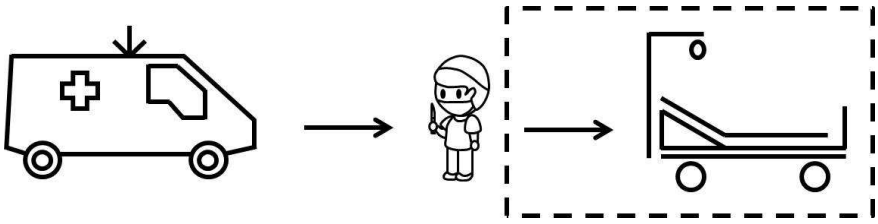
Figure 5.14: Results OR assistants and pathology survey

Logistic department

From an interview with the logistic department we learned that logistic employees do not want surgeries to be swapped. The logistic department assigned five points, the maximum amount, to swapping two surgeries.

Part IV

Ward planning



Chapter 6

Reducing the number of required beds by rearranging the OR-schedule

6.1 Introduction

The starting point of this research was a request from HagaZiekenhuis to get more insight in the factors that influence the bed occupancy. The operating room (OR) schedule is one of the most important factors that influence the bed occupancy, since most of the surgical patients have to be admitted at one of the wards after surgery. Therefore, it is important to consider the required bed capacity when creating the OR-schedule, which is the topic of this chapter. First, we analyze this problem and investigate several approaches to solve it, and second, we show what improvements can be made in HagaZiekenhuis.

There is a vast amount of literature on OR planning and scheduling. Cardoen et al. [18] and Hulshof et al. [51] provide an overview of papers that address this topic. Several of the papers mentioned by [18] and [51] address the issue of considering the wards when creating an OR-schedule. The first paper that considers this topic is the work of Beliën and Demeulemeester [7]. They schedule blocks of elective surgeries of the same type by assigning them to a day in the planning horizon while minimizing the number of required beds. They assume that the length of stay (LOS) is given by a multinomial distribution that differs per surgery type. The number of required beds resulting from an OR-schedule is approximated in several ways, however, no exact formulation is used. Beliën et al. [8] extend this approach by including multiple wards instead of one, allowing different block lengths, and by scheduling individual surgeons instead of surgeon groups. In addition, they develop a decision support system that visualizes the OR-schedule and the resulting bed occupancy.

Van Oostrum et al. [93] schedule surgical procedures instead of OR-blocks by assigning the procedures to an OR and to a day in the planning horizon. The LOS of the patients is assumed to be deterministic and by using this deterministic LOS, the number of required beds is minimized. Adan et al. [3], [4] schedule surgical procedures by assigning them to a day in the planning horizon as is done by Van Houdenhoven et al. [91] and Van Oostrum et al. [93]. But opposite to [91] and [93], they assume that a fixed number of beds is available at the hospital and minimize the over- and underutilization of these beds. Thus, the number of required beds is not minimized, but their use is optimized.

Chow et al. [20] develop an Integer Linear Programming (ILP) model to generate improved OR-schedules in terms of the maximum expected bed occupancy. This expected bed occupancy is calculated by using the expected LOS of surgery types, and after this, the average bed occupancy per day is determined by means of simulation.

As in [7], Vanberkel et al. [94], [95] schedule blocks of surgeries of the same type and assume a multinomial distribution of the LOS that differs per surgery type. However, in contrast to [7], they analytically determine the complete probability distribution of the number of occupied beds for each day in the planning horizon. The goal of this approach is not to minimize the number of required beds, but to develop a model that can be used as an evaluation tool for the OR-schedule.

Bekker and Koeleman [6] developed a method that determines the mean bed occupancy per day when a weekly admission pattern is given. In addition, they use a quadratic programming model to determine the optimal number of elective admissions per day such that an average desired bed occupancy per day is achieved.

For the operational planning level, Cardoen et al. [16], [17] propose a mixed integer linear programming approach and a column generation approach to determine the sequence in which patients must have surgery on a given day such that the peak use of recovery beds is minimized. They assume the LOS on the recovery to be deterministic, and thus, no stochasticity is included. Fei et al. [33] also focus on the sequence in which patients must have surgery, however, they consider the number of beds available at the recovery to be fixed, and therefore, do not focus on minimizing the peak use.

Many of the discussed papers consider the expected LOS of patients instead of the LOS probability distribution or focus on minimizing the maximum expected bed occupancy without considering the bed occupancy probability distribution. However, in practice, the LOS of patients is stochastic, and thus, it is important to also consider the variance in the bed occupancy. In this chapter, we incorporate both the stochasticity of the LOS and of the bed occupancy to account for these variances. As in practice most hospitals use a cyclic OR-schedule, we develop an OR-schedule by assigning OR-blocks to a day in the planning horizon. We assume that an OR-block consists of not only one but several surgical procedure types to make the problem more suitable for application. Because we schedule surgery types and not individual patients, this scheduling problem is considered to be on the tactical level. As in [7] and [95], we assume the LOS to be multinomially distributed. This distribution can easily be obtained from historical data. We use the analytical formulation of Vanberkel et al. [95] to determine the number of required beds and minimize this number while taking into account several restrictions on the OR-schedule such as OR, surgeon, and instrument availability. As the problem originated in HagaZiekenhuis, we focus on resource constraints that are relevant in the setting of this hospital. However, it is possible to add additional constraints without destroying the structure of the developed model. Note that we only consider the scheduling of elective surgeries, but it is quite easy to also

include emergency surgeries when determining the number of required beds.

The developed model is discussed in Section 6.2 and it consists of linear constraints and a complex non-linear objective function that involves the convolution of discrete distributions. To deal with this complexity, we introduce in Section 6.3 two different approaches to approximate an optimal solution. The first approach is a local search approach that takes the complex formulation of the objective function into account. We have chosen to use Simulated Annealing (SA) since this approach is easy to implement and has proven to be successful for other combinatorial optimization problems. The second approach reduces the complexity of the problem by linearizing the objective function. Although we prove the resulting problem to be $\mathcal{NP}$-hard, we model and solve the resulting problem as an ILP, because our considered instances are small enough to be solved within a reasonable amount of time. By comparing these two different approaches, we can determine whether it is better to not fully search the solution space with a complete evaluation of the objective function or to approximate the objective function and search the complete solution space. In fact, it is investigated whether it is necessary to model the problem in full detail to be able to achieve a good solution.

The comparison is performed on data provided by HagaZiekenhuis. However, we use this data not just for comparing the two contrasting approaches, but we also aim to support HagaZiekenhuis by determining which resources are a bottleneck for minimizing the number of required beds. We do this by considering several what-if scenarios that relax some or all of the resource constraints. The computational results of the comparison and the what-if scenarios are given in Section 6.4. Section 6.5 presents conclusions and gives recommendations for further research.

6.2 Problem formulation

Hospitals aim to use as few beds as possible. When less beds are used, as a consequence less personnel is needed and less money is spent on cleaning and maintaining these beds. Another effect of using less beds is that also the bed occupancy during the week is better leveled and this reduces stress on the wards.

In hospitals, the number of beds occupied during the week is mostly determined by the OR-schedule. In general, a patient is admitted on the day of surgery, and after surgery, the patient must stay in the hospital for a few extra days. Thus, in order to influence the number of beds used, we should create an OR-schedule that minimizes the number of required beds, and thereby, levels the number of occupied beds as much as possible. HagaZiekenhuis, like many other hospitals, uses a cyclic OR-schedule that repeats every $\mathcal{T}$ days. This means that we have to develop such a cyclic OR-schedule for $\mathcal{T}$ days and not an OR-schedule for a whole year.

An OR-schedule consists of OR-blocks that are assigned to days of the planning cycle. Each OR-block is dedicated to a specific specialism or specialist and is filled with several surgery types chosen by this specialism or specialist. Thus, each specialism or specialist provides a list containing as many OR-blocks as this

specialism or specialist gets during a period of $\mathcal{T}$ days. It only remains to assign these OR-blocks to a specific day in the planning horizon to create an OR-schedule.

6.2.1 Restrictions

In this section, we discuss several restrictions on the OR-schedule that are relevant for HagaZiekenhuis. Although we only provide these specific constraints, it is possible to add additional constraints without destroying the structure of the chosen approach.

Let K be the given set of OR-blocks. To each OR-block $k \in K$ we have to assign a specific day $t \in T = \{1, \ldots, \mathcal{T}\}$. For this, we define binary decision variables X_{kt} that are one when OR-block $k \in K$ is assigned to day $t \in T$ and zero otherwise. Then, the following constraints ensure that all OR-blocks $k \in K$ are assigned to a day in the OR-schedule:

$$\sum_{t \in T} X_{kt} = 1, \qquad \forall k \in K. \tag{6.1}$$

The assignment of OR-blocks to days is limited by several constraints. First, some OR-blocks can only be performed in a subset of the available ORs, because, for example, special equipment is needed that is not available in all ORs. To model this, we define a set J of different OR types and for OR type $j \in J$, we denote by the subset $K_j \subseteq K$ the OR-blocks that can be performed in OR type $j \in J$. In addition, the number of available ORs of type $j \in J$ on day $t \in T$ is limited and denoted by a_{jt}. The following constraints ensure that the assignment of OR-blocks to days fulfills these limitations:

$$\sum_{k \in K_j} X_{kt} \leq a_{jt}, \quad \forall j \in J, t \in T. \tag{6.2}$$

Each OR-block is allocated to a specific surgeon type, because most surgeons in HagaZiekenhuis are specialized in a certain set of surgery types. The surgeon types are given by set S and the OR-blocks that have to be performed by surgeon type $s \in S$ are given by subset $K_s \subseteq K$. The number of available surgeons of type $s \in S$ on day $t \in T$ is limited and denoted by b_{st}. The following constraints ensure that the assignment of OR-blocks to days fulfills these restrictions:

$$\sum_{k \in K_s} X_{kt} \leq b_{st}, \quad \forall s \in S, t \in T. \tag{6.3}$$

Each OR-block consists of several surgeries that must be performed consecutively. The total set of possible surgery types is defined by I and the number of surgeries of a specific type $i \in I$ performed in OR-block $k \in K$ is given by o_{ik}. For each surgery type $i \in I$ a specific set of instruments is needed to perform the

surgery. The set of all available instrument sets is given by set R and w_{kr} denotes how many instrument sets $r \in R$ are needed for OR-block $k \in K$. Because a limited number of instrument sets is available and the instrument sets have to be sterilized after surgery, the number of surgeries that need instrument set $r \in R$ scheduled per day is limited by q_r. This is ensured by the following constraints:

$$\sum_{k \in K} X_{kt} w_{kr} \leq q_r, \qquad \forall r \in R, t \in T. \tag{6.4}$$

Note that the o_{ik} values are not used explicitly, but are covered in the w_{kr} values. However, we have introduced the values since they are needed in the next section.

6.2.2 Objective function

Constraints (6.1)-(6.4) are the restrictions on the decision variables X_{kt}, and therefore, describe the set Φ of feasible solutions. In this section, we specify the quality of a feasible solution $\phi \in \Phi$ given by the maximum number of beds needed during the entire planning horizon. To determine this number for a proposed OR-schedule, we have to determine the bed occupancy for each day. If we would specify the bed occupancy by a deterministic measure (e.g., maximum or expected number of used beds), we do not take the stochastic nature of the LOS into account. Using the expected bed occupancy per day results in canceling patients for surgery because quite often not enough beds are available to admit them after surgery. Using the maximum number of beds needed leads to a solution for which almost always too many beds are available. Therefore, we choose to calculate the complete bed occupancy probability distribution per day and afterwards take the p-percentile of these probability distributions to ensure that sufficient beds are available with p percent chance. Since these percentiles represent the number of beds needed on day $t \in T$ of the planning horizon, we obtain the number of beds needed in the wards by taking the maximum over all days.

For a given OR-schedule, the probability distribution of the bed occupancy can be obtained by using the LOS distribution of all surgery types scheduled in the OR-blocks. The LOS distribution of each surgery type $i \in I$ is given by a multinomial distribution that can be obtained from historical data. After the LOS distributions are obtained from the historical data, we can compute the probability distribution of the bed occupancy for each day as in [95] by taking discrete convolutions of the LOS distributions. In the following paragraphs, we shortly explain this method. For a more detailed description of this method, we refer to Vanberkel et al. [95].

Before we explain the method into more detail, we want to note that the LOS of a patient in the historical data might be influenced by the bed occupancy on the wards. When the wards are more crowded, it is likely that a patient is discharged sooner than would have been the case when the ward was less crowded. However, because our approach is on the tactical level and not on the operational level, we do not include these dependencies.

The probability distribution of the LOS of surgery type $i \in I$ is given by values l_n^i, which denote the probability that the LOS of a surgery type $i \in I$ is exactly n days ($n \in \{1, \ldots, \mathcal{L}_i\}$), where $\mathcal{L}_i$ is the maximum LOS of surgery type $i \in I$. From this, we can determine the conditional probability that a patient who is still admitted on day n is discharged that day, which is denoted by d_n^i. Note that d_1^i denotes the probability that a patient is discharged on the day of surgery (i.e., an outpatient surgery) and $d_{\mathcal{L}_i}^i = 1$. The value of d_n^i is given by

$$d_n^i = \frac{l_n^i}{\sum_{m=n}^{\mathcal{L}_i} l_m^i}. \tag{6.5}$$

From these values, we can calculate the probability distribution $h_n^{ik}(x)$ that n days after carrying out OR-block $k \in K$, x patients of surgery type $i \in I$ are still in recovery. Recall that o_{ik} denotes the number of patients of type $i \in I$ assigned to OR-block $k \in K$. Therefore, these probabilities are computed recursively as follows:

For $n = 1$:

$$h_1^{ik}(x) = \begin{cases} 1 & \text{when } x = o_{ik}, \\ 0 & \text{otherwise.} \end{cases} \tag{6.6}$$

For $n > 1$:

$$h_n^{ik}(x) = \sum_{y=x}^{o_{ik}} \binom{y}{x} \left(d_{n-1}^i\right)^{y-x} \left(1 - d_{n-1}^i\right)^x h_{n-1}^{ik}(y). \tag{6.7}$$

Next, we take discrete convolutions of $h_n^{ik}(x)$ over all $i \in I$ to determine the bed occupancy caused by OR-block $k \in K$. This gives the probability $\tilde{h}_n^k(x)$ that n days after carrying out OR-block $k \in K$, x patients are still in recovery:

$$\tilde{h}_n^k(x) = h_n^{1k}(x) * h_n^{2k}(x) * \ldots * h_n^{\mathcal{I}k}(x). \tag{6.8}$$

Because we use a cyclic OR-schedule, which repeats every $\mathcal{T}$ days, patients who had surgery in one cycle may still be admitted in the next cycle. Therefore, we must take into account $\lceil \mathcal{N}_k / \mathcal{T} \rceil$ consecutive cycles, where $\mathcal{N}_k$ denotes the maximum LOS of the surgeries scheduled in OR-block $k \in K$, i.e, $\mathcal{N}_k = \max_{i \in I | o_{ik} \geq 1} \mathcal{L}_i$. In other words, $\mathcal{N}_k$ represents the range of one cycle of the OR-schedule. Now, by again using discrete convolutions, we can compute the probability distribution $H_t^k(x)$ of recovering patients on day $t \in T$ of the cycle induced by OR-block $k \in K$ as follows:

$$H_t^k(x) = \tilde{h}_t^k(x) * \tilde{h}_{t+\mathcal{T}}^k(x) * \tilde{h}_{t+2\mathcal{T}}^k(x) * \ldots * \tilde{h}_{t+\lceil \mathcal{N}_k/\mathcal{T} \rceil \mathcal{T}}^k(x). \tag{6.9}$$

The last step in calculating the probability distribution of the bed occupancy is to combine the probability distributions $H_t^k(x)$ for all OR-blocks. To do this, we first have to shift the distribution $H_t^k(x)$ such that the patients who have surgery in OR-block $k \in K$ are admitted on the day they have surgery, i.e., the day $t \in T$ for which $X_{kt} = 1$. The shifted probability distribution is denoted by $\bar{H}_t^k(x)$ and is defined as follows:

$$\bar{H}_t^k(x) = \begin{cases} H^k_{t-\hat{t}+1}(x) & \text{for } \hat{t} \text{ with } X_{k\hat{t}} = 1 \text{ and } \hat{t} \leq t, \\ H_{t-\hat{t}+\mathcal{T}+1} & \text{otherwise.} \end{cases} \tag{6.10}$$

By taking the discrete convolutions of $\bar{H}_t^k(x)$ over $k \in K$, we now determine the probability distribution of the bed occupancy for each day $t \in T$ denoted by H_t, which is computed by

$$H_t(x) = \bar{H}_t^1(x) * \bar{H}_t^2(x) * \ldots * \bar{H}_t^{\mathcal{K}}(x). \tag{6.11}$$

Thus, the number of required beds $\gamma(\phi)$ for a given solution $\phi \in \Phi$ is given by:

$$\gamma(\phi) = \max_{t \in T} \min \left\{ x \,\middle|\, \sum_{y=0}^{x} H_t(y) \geq \frac{p}{100} \right\}. \tag{6.12}$$

Note that the calculations for (6.5)-(6.9) can be performed beforehand. Nevertheless, determining the objective function for a new OR-schedule still involves convoluting several probability distributions as shown by equation (6.11). Therefore, it is not straightforward to quantify or predict the effect in the objective function when changing an OR-schedule and it is hard to approximate the objective function. Moreover, calculating the objective function takes a lot of computation time. To reduce this computation time, we can either choose to not fully search the solution space or to approximate the objective function. This leads to the following two solution approaches: (1) use a local search heuristic based on the given constraints and objective function and (2) approximate the objective function and incorporate this approximation in an ILP that includes the given constraints of the OR-schedule. For the second approach, the original objective value is determined afterwards to determine the number of beds needed in practice and to make a fair comparison between the two approaches. The comparison is used to determine whether it is better to not fully search the solution space with a complete evaluation of the objective function or to approximate the objective function and search the complete solution space. The two approaches are discussed in more detail in the following section.

6.3 Solution methods

Because of the complex objective function, we cannot solve reasonable sized instances of the problem to optimality. Therefore, we have to make a choice between using a heuristic procedure to solve the original problem and using a global approach to solve a simplified version of the problem. Both approaches do not guarantee to find an optimal solution, therefore, we want to investigate which of these two methods leads to better solutions. The first approach is based on SA, which is a local search method. The second approach is an ILP that uses an approximation of the objective function. In the following, these two approaches are discussed in more detail.

6.3.1 Local search approach: simulated annealing

As a first approach to solve our problem we have chosen SA [56]. SA is a local search procedure that in each step moves from the current solution, denoted by ϕ_c, to a randomly selected neighbor solution, denoted by ϕ_n. A solution is represented by the assignment of OR-blocks to a day in the planning horizon and is considered to be feasible when it satisfies constraints (6.1)-(6.4). As neighbor solutions, we consider all feasible solutions that can be obtained by swapping two OR-blocks that are assigned to two different days. We do not consider swapping two OR-blocks assigned to the same day, because this does not affect the objective value. If the randomly selected neighbor solution has a lower objective function value than the current solution, i.e., $\gamma(\phi_n) \leq \gamma(\phi_c)$, the neighbor solution is accepted as the new current solution. Otherwise, the neighbor solution is accepted with a probability that depends on the objective value of the current and neighbor solution and on a temperature parameter. This temperature parameter, denoted by Γ, gradually decreases during the search process. For each temperature value, we perform ω iterations that together form a Markov chain, because the next state only depends on the current state. Also, during the entire process of SA, we keep track of the best solution found so far. See Section 2.4.2 for a more detailed description of this method.

Summarizing, our implementation of SA is as follows, where $\bar{\phi}$ denotes the current best solution:

Step 1. Start with the initial solution ϕ_c given by the OR-schedule currently used at the hospital. Set $\bar{\phi} := \phi_c$ and determine the objective function value $\gamma(\phi_c)$. Set the initial temperature, i.e., $\Gamma := \Gamma_s$ and a reduction factor α.

Step 2. Repeat ω times:

- Step (a) Randomly select a neighbor solution ϕ_n of the current solution and determine $\gamma(\phi_n)$.
- Step (b) If $\gamma(\phi_n) \leq \gamma(\phi_c)$, set $\phi_c := \phi_n$, and if $\gamma(\phi_n) \leq \gamma(\bar{\phi})$, set $\bar{\phi} := \phi_n$. Otherwise, set $\phi_c := \phi_n$ with probability $e^{\frac{\gamma(\phi_c)-\gamma(\phi_n)}{\Gamma}}$.

Step 3. Set $\Gamma = \alpha\Gamma$. If $\Gamma < \Gamma_f$, the final temperature, then stop. Else, go to 2.

We choose the initial temperature Γ_s such that an increase of the objective value at the beginning of the procedure is accepted with a relatively high probability. This is needed to easily escape from a local minimum. We observe that the maximum increase of the objective value equals the maximum over the number of surgeries assigned to an OR-block minus the minimum over the number of surgeries assigned to an OR-block, i.e., $\max_{k \in K} \sum_{i \in I} o_{ik} - \min_{k \in K} \sum_{i \in I} o_{ik}$, because all patients are admitted on the day of surgery. We want to accept this maximum increase at the start of the procedure with probability 0.95, thus the initial temperature is given by:

$$\Gamma_s = \frac{\max_{k \in K} \sum_{i \in I} o_{ik} - \min_{k \in K} \sum_{i \in I} o_{ik}}{\ln 0.95}. \tag{6.13}$$

Using the same approach, we determine the final temperature Γ_f. This temperature is chosen such that the probability of accepting the minimum increase of the objective value is very low. This means that at the end of the procedure hardly any worse solutions are accepted, and thus, the procedure converts to a local minimum. Since our objective function returns an integer number of beds, the minimum increase is one bed. Thus, we set the threshold temperature Γ_f such that an increase of one bed is accepted with probability 0.001, i.e.,

$$\Gamma_f = \frac{-1}{\ln 0.001}. \tag{6.14}$$

We set the number of iterations for each temperature value equal to the number of neighbor solutions that can be achieved by one swap of the current solution. This number is equal to the total number of OR-blocks, i.e, ω equals the cardinality of set K, because in theory each OR-block can be swapped with one of the other OR-blocks. However, due to the restrictions described in Section 6.2.1, some swaps are prohibited. The reduction factor α is set to 0.95.

During preliminary runs, we tested the effect of varying the values for the initial temperature Γ_s, the final temperature Γ_f, the length of the Markov chain ω, and the reduction factor α. We evaluated different values for the initial temperature by determining the performance of the SA approach when the acceptance probability of the maximum increase is set to 0.75 and 0.5. This means that we limit the increase in the objective function at the start of the SA approach. The final temperature is varied by increasing the acceptance probability of the minimum increase to 0.1, 0.01, and 0.005. When the computational time of the SA approach is increased, the approach might provide better results. Increasing the computational time can be achieved by increasing the length of the Markov chain or by increasing the reduction factor. For the Markov chain, we have evaluated the values 100 and 500 and for the reduction factor the value 0.97. To further investigate the effect of the reduction factor, we have also evaluated the performance of the SA approach when the reduction factor is set to 0.9. None of these mentioned values lead to a significant change of the performance of the SA approach. Only when all parameters are set such that the computing time is maximal, i.e., Γ_s and Γ_f are set at their original values and ω and α are set to 500 and 0.97, respectively, the performance improves slightly. However, when using these parameter settings the computational time increases by a factor 20. Therefore, we use the values of the parameters as described above.

6.3.2 Global approach: linearization of objective function

The local search approach described in the previous section incorporates the complete evaluation of the objective function but only searches a part of the solution space. The global approach described in this section searches the entire solution

space, but for such an approach the relation between a solution and the objective function must be evident. However, Section 6.2 shows that there is no straightforward and direct relation between a given OR-schedule and the resulting required number of beds. Therefore, we choose to linearize the objective function by replacing it with the maximum over the expected number of occupied beds per day.

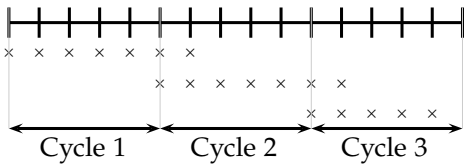


Figure 6.1: Overlap of patients' multiple cycles

For calculating the expected number of occupied beds per day, we follow the approach of Beliën and Demeulemeester [7]. However, their formula does not work properly when the LOS of a patient exceeds the planning horizon. Figure 6.1 shows that when the maximum LOS $\mathcal{L}_i$ of a surgery type $i \in I$ exceeds the length $\mathcal{T}$ of the planning horizon, patients of two cycles of the OR-schedule may be admitted simultaneously, i.e., patients from different cycles may overlap. In [7], it is assumed that this holds for all days in the planning horizon, however, this only holds for a few days as shown in Figure 6.1, where a situation is sketched with $\mathcal{T} = 5$ and $\mathcal{L}_i = 7$. This deficiency can be accounted for by a small modification in the weight factor used in the formula defined in [7]. This modification ensures that patients are only counted multiple times on the days of the planning horizon that the LOS of several cycles overlap. As a result, the expected number of occupied beds $\gamma_t(\phi)$ on day $t \in T$ of the planning horizon is given by:

$$\begin{aligned}\gamma_t(\phi) \;=\; & \sum_{i\in I}\sum_{k\in K}\sum_{\tau\le t}\left(\sum_{n=t-\tau+1}^{\mathcal{L}_i} l_n^i o_{ik}\left\lceil\frac{n-t+\tau}{\mathcal{T}}\right\rceil\right)X_{k\tau} \\ & +\sum_{i\in I}\sum_{k\in K}\sum_{\tau>t}\left(\sum_{n=\mathcal{T}+t-\tau+1}^{\mathcal{L}_i} l_n^i o_{ik}\left\lceil\frac{n-\mathcal{T}-t+\tau}{\mathcal{T}}\right\rceil\right)X_{k\tau}. \qquad (6.15)\end{aligned}$$

Equation (6.15) determines for each day t in the planning horizon, the impact of all OR-blocks on the bed occupancy. Thus, for all OR-blocks it is determined whether patients operated on in this OR-block are still admitted in the hospital while taking into account overlapping cycles.

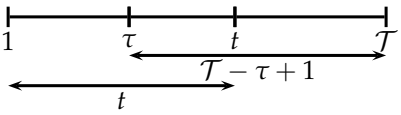


Figure 6.2: Situation for first part of equation (6.15)

The first part of the equation only considers patients who are operated before or on the considered day $t \in T$ by integrating the binary variable $X_{k\tau}$, which is one when OR-block $k \in K$ is scheduled on day $\tau \in T$ and summing over all $\tau \leq t$. The expected number of patients of type i operated in block k and still admitted n days after surgery is given by $l_n^i o_{ik}$. As we only want to include the patients that are still admitted on day t, we only include l_n^i for $n \in \{t-\tau+1, \ldots, \mathcal{L}_i\}$. A patient should be counted only once if his/her LOS lies between $t-\tau+1$ and $\mathcal{T}-\tau+t$ and counted twice if his/her LOS lies between $\mathcal{T}+t-\tau+1$ and $2\mathcal{T}-\tau+t$ etc., which is represented by $\left\lceil \frac{n-t+\tau}{\mathcal{T}} \right\rceil$. Figure 6.2 depicts the minimum LOS of a patient that should be counted twice when determining the expected number of occupied beds.

The second part of the equation only considers patients who are still admitted in one of the cycles after the cycle in which they had surgery. This means that we only include patients who have a LOS of $(\mathcal{T}-\tau+1)+(t-1+1) = \mathcal{T}-\tau+1+t$ or more days. This minimum LOS is depicted in Figure 6.3. This means that when the patient's LOS is between $\mathcal{T}-\tau+1+t$ and $2\mathcal{T}-\tau+t$, the patient should only be counted once. When the LOS is between $2\mathcal{T}-\tau+t$ and $3\mathcal{T}-\tau-1+t$, the patient should be counted twice. This is represented by $\left\lceil \frac{n-\mathcal{T}-t+\tau}{\mathcal{T}} \right\rceil$.

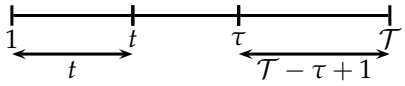


Figure 6.3: Situation for second part of equation (6.15)

Note that the expected value associated with the probability distribution of the bed occupancy as given in Section 6.2.2 corresponds to $\gamma_t(\phi)$. We can incorporate the linearized objective function given by equation (6.15) in an ILP that includes the constraints given in Section 6.2.1. Then, the resulting ILP is:

$$\begin{aligned} \min_{\phi \in \Phi} \quad & \bar{\gamma}(\phi) && \\ \text{s. t.} \quad & \text{(6.1)-(6.4), (6.15)}, && \\ & \bar{\gamma}(\phi) \geq \gamma_t(\phi), && \forall t \in T, \\ & X_{kt} \in \{0,1\}, && \forall k \in K, t \in T. \end{aligned} \tag{6.16}$$

This resulting problem is strongly $\mathcal{NP}$-hard. For this, consider an instance with three ORs, a planning horizon of $\mathcal{T}$ days, and thus, $3\mathcal{T}$ OR-blocks. Each of the $3\mathcal{T}$ OR-blocks consists of a_k patients with $k = 1, \ldots, 3\mathcal{T}$ and $\sum_{k \in K} a_k = \mathcal{T}b$, where b is a given positive integer. In addition, each of the patients has a LOS of exactly one day. For this instance, determining whether there exists an OR-schedule that requires b beds is equivalent to determining whether there are $\mathcal{T}$ pairwise disjoint subsets $R_l \subset \{1, \ldots, 3\mathcal{T}\}$ such that $\sum_{R_l \in K} a_k = b$ for $l = 1, \ldots, \mathcal{T}$, which is known as the 3-partition problem [36].

The ILP given by (6.16) consists of $|K|\mathcal{T}$ binary variables and $\big(|J| + |S| + |R| + 2\big)\mathcal{T} + |K|$ constraints. After solving the ILP, each OR-block is assigned to a day in the planning horizon leading to a solution $\phi \in \Phi$. For this solution, the real objective value, i.e., the maximum over the p-percentiles of the resulting bed occupancy probability distribution, can be determined using the method described in Section 6.2.2.

The ILP only provides an optimal solution to the original problem when for each pair of solutions $\phi, \phi' \in \Phi$ the following holds: $\bar{\gamma}(\phi) \leq \bar{\gamma}(\phi') \Leftrightarrow \gamma(\phi) \leq \gamma(\phi')$, i.e., the ordering of the solutions in set Φ according to the expected number of beds should be the same as the ordering according to the number of beds needed for the p-percentile. In general, the validity of this relation depends on the input data of the LOS distributions and cannot be guaranteed for all pairs of solutions. However, if it holds for most pairs, good solutions to the original problem may be obtained by solving the ILP.

6.4 Computational results

The purpose of this section is twofold. First, we compare the SA and ILP approach in Section 6.4.1 for 100 random generated instances based on data from HagaZiekenhuis. For the global approach, the original objective value for the resulting OR-schedule is determined afterwards such that a fair comparison can be made between the local and global approach. The results are used to determine whether it is better to not fully search the solution space with a complete evaluation of the objective function or to approximate the objective function and search the complete solution space. Second, we consider several what-if scenarios for HagaZiekenhuis with the solution approach that performed best in Section 6.4.1. We use these scenarios to determine whether the resource availability in HagaZiekenhuis limits the reduction of the number of required beds.

As mentioned in the introduction, the goal of the research is to give HagaZiekenhuis more insight in the factors that influence their bed occupancy. Therefore, the data used in the following sections is based on data of HagaZiekenhuis. HagaZiekenhuis provided us with an OR-schedule of the orthopedics department with a planning horizon of 28 days, where up to three ORs and nine surgeons are available. The exact availability of the ORs and surgeons is given for each day in the planning horizon. The OR-schedule consists of 49 unique OR-blocks that have to be scheduled exactly once during the planning horizon. In total, 43 different surgery types are scheduled and the LOS per surgery type varies from 1 to 59 days with an average LOS of 3.7 days. For each surgery type, it is denoted which instrument sets are needed and for each of the ten available types of instrument sets it is given how many are available each day. As the number of required beds, we take the maximum of the 95-percentile of the probability distribution of the bed occupancy over the 28 days.

6.4.1 Comparing local and global approach

To determine which of the two considered approaches performs better, we have generated 100 random instances based on the data of HagaZiekenhuis. For the original data set, each OR-block had to be performed exactly once. To get different instances having similar characteristics as the original data, we vary the number of times a certain OR-block has to be performed during the planning horizon, i.e., some OR-blocks are not performed at all and some are performed multiple times in one cycle. To make sure there exists a solution that satisfies the fixed OR, surgeon, and instrument sets availabilities, we generate the instances as follows: first, we randomly select for each available OR for each day in the planning horizon a surgeon available on that day. After this, we randomly select one of the OR-blocks that can be performed in the considered OR by the selected surgeon. During this selection process, we also consider the availability of instrument sets. Because the number of times an OR-block is performed varies for the generated instances, we create instances that vary among the number of surgeries and the average LOS of the patients. Therefore, we analyze a broad range of different instances. The number of surgeries varies between 201 and 236 and the average LOS over all surgeries scheduled in a random instance varies between 3.75 and 4.11 days.

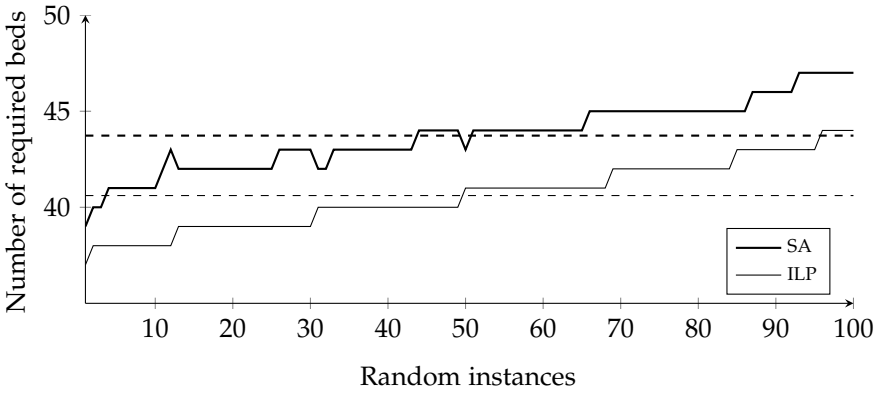


Figure 6.4: Results 100 random instances for 95-percentile

The SA approach is implemented in CodeGear Delphi and the ILP is solved with CPLEX 12.3. Both methods are executed on an Intel Core2 Duo CPU P8600 2.40 GHz with 3.45 GB RAM. Since proving the optimality of a solution by CPLEX takes quite some time, we interrupt the solver after ten minutes. The maximum achieved integrality gap for the 100 random instances is 1.41%. The results for the 100 instances can be found in Figure 6.4 where the dashed lines denote the average of the objective values for the two approaches. Note that the random instances are sorted according to the objective function values to clarify the differences between the two approaches.

Figure 6.4 shows that the global approach performs better than the local search

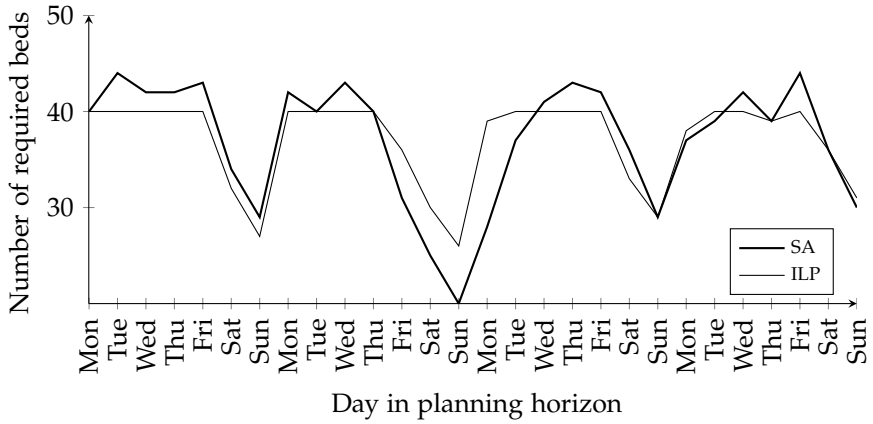


Figure 6.5: Difference in leveled bed occupancy

approach for all of the 100 random instances. In addition, we see that the difference between the global and local objective value is almost everywhere the same. The difference in the objective values is two beds for 8% of the instances, three beds for 73% of the instances, four beds for 18% of the instances, and five beds for 1% of the instances. Note that both objective values represent the maximum of the 95-percentile of the probability distribution of the number of required beds over all days.

Figure 6.5 shows one of the random instances for which the difference between the two approaches can be explained nicely. The peaks for the global approach are flat, which results in a constant number of occupied beds. This flat bed occupancy can be achieved because the OR-schedule leaves enough room for improvement. The constraints do not restrict the solution that much that a flat bed occupancy cannot be achieved. The peaks of the local search approach fluctuate, which results in a higher number of required beds. Note that because of these fluctuations, only 20 patients are admitted on the second Sunday of the cycle. When the peaks in bed occupancy are decreased, the number of admitted patients on this day would likely increase.

The solution time needed for the local search approach varies between 32 and 74 seconds with an average of 42 seconds. The solution time needed for the global approach is set to 600 seconds. Therefore, as expected, the global approach takes longer than the local search approach, but ten minutes is still a reasonable amount of time.

The ILP also outperforms the SA approach when we look at the 90-percentile and the 85-percentile. As the objective function of the ILP only depends on the expected number of beds needed, we do not have to solve the ILP again, but only have to determine the 90-percentile and 85-percentile for the solutions found by the ILP. The SA does consider the chosen percentile during the procedure, and

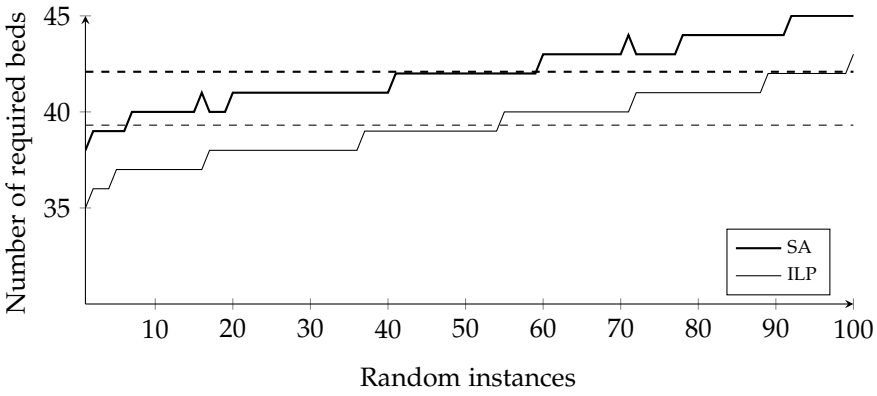


Figure 6.6: Results 100 random instances for 90-percentile

therefore, we have to run SA again to determine new solutions for these percentiles. The results for the 100 random instances are depicted in Figures 6.6 and 6.7.

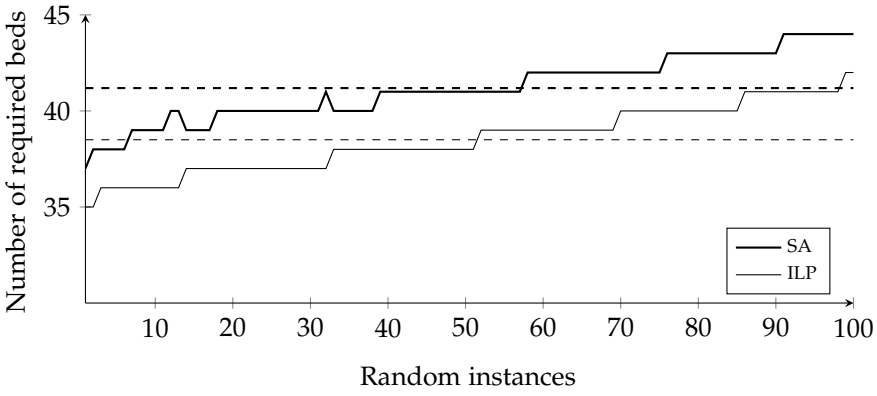


Figure 6.7: Results 100 random instances for 85-percentile

The given results already indicate that for the considered instances based on the characteristics of the data from HagaZiekenhuis, it is not necessary to include the detailed objective function to determine a good OR-schedule as the ILP provides better results than the SA approach. As stated in Section 6.3.2, the ILP provides an optimal solution to the original problem when the ordering of the solutions in set Φ according to the expected number of beds is the same as the ordering according to the number of beds needed for the p-percentile. To investigate to which extent this holds for the considered instances, we have plotted the expected number of beds, the 85-percentile, and the 95-percentile of beds needed in Figure 6.8 for 1,000 random solutions to the original instance provided by HagaZiekenhuis. Note that

the random solutions are sorted in increasing order of the expected number of beds needed.

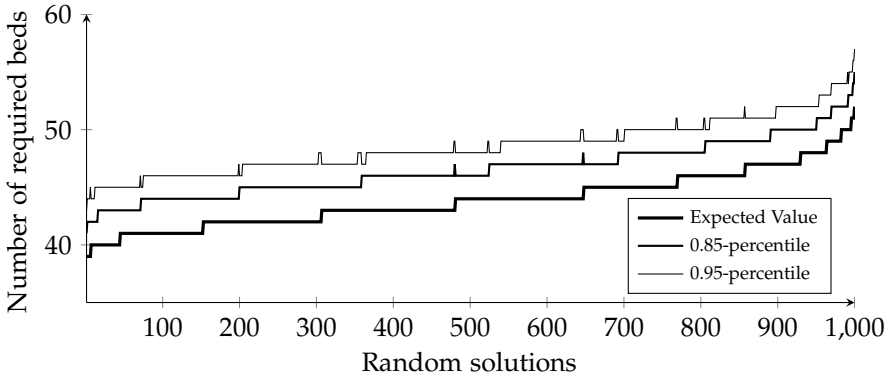


Figure 6.8: Relation expected value, 0.85-percentile, and 0.95-percentile of 1,000 random solutions for instance based on data HagaZiekenhuis

The wiggliness of the graphs of the 85-percentile and 95-percentile shows that the stated property does not hold for all 1,000 solution pairs. However, the small number of wiggles indicate that this property does hold for most pairs. This gives an explanation why the ILP provides good solutions to the original problem. Even though we cannot guarantee that the ILP finds an optimal solution, the ILP performs better than the SA approach when we compare both methods in the ability to reach a good feasible solution for the considered instances. Figure 6.8 shows that the error made by using the expected number of beds instead of the 95-percentile is at most two beds for the considered 1,000 random solutions. If we, for example, choose one of the solutions with expected value 39, the 95-percentile varies from 43 to 45. Thus, although two solutions may be considered to be equally good based on the expected value, one of the solutions might outperform the other when considering the 95-percentile. However, if this error is limited to two beds, the ILP still provides good solutions to the original problem if only the expected values are used.

Although the ILP performs well for the considered instances, we cannot guarantee that this method also works on instances from other hospitals. For each considered setting, first the relation between the expected value and the chosen p-percentile should be investigated. When the property stated above holds for most solution pairs, the ILP can be used to find good solutions to the original problem. Else, it might be better to use the SA approach to solve the problem. For instances arising from practice, we believe that the ILP will outperform the SA approach as we expect that the LOS distributions will not differ much from the LOS distributions used in the instance provided by HagaZiekenhuis. Nevertheless, it would be interesting to determine what conditions would result in always or at least regularly fulfilling the stated property. These conditions can then be used to determine

beforehand which of the two proposed methods would be most suitable. However, determining these conditions is outside the scope of this research.

6.4.2 What-if scenarios

The starting point of this research was the request from HagaZiekenhuis to get more insight in the factors that influence the bed occupancy. Therefore, we use the global approach to show the reduction in the number of required beds when the OR-schedule is changed and we investigate whether the resource availability at HagaZiekenhuis limits this reduction. The hospital provided us an OR-schedule with a planning horizon of 28 days used by the orthopedics department. For the OR-schedule as provided by HagaZiekenhuis, 48 beds were needed to admit all surgical patients. We determined a new OR-schedule by solving the ILP and interrupting the solver after ten minutes. To determine whether one or more of the constraints limit the improvement of the OR-schedule, we also consider the following scenarios:

- **Relax the number of available ORs per day**: The number of ORs of type $j \in J$ that are available each day is given by a_{jt}. By relaxing constraint (6.2), we do not restrict the model to schedule a fixed number of OR-blocks per day. Since we relax the problem, we expect to come up with a schedule that requires less beds on the wards. We allow a maximum of five OR-blocks scheduled per day since five ORs are physically available at the operating department. Note that the number of available ORs in the weekend is still set to zero as usually no surgeries are performed during this time.

- **Relax the surgeon availability**: As with the previous scenario, the surgeon availability corresponds to a constraint in the model. To determine what restriction this constraint imposes on the resulting OR-schedule, and thus, on the required number of beds, we solve the model while relaxing constraint (6.3). Note that the surgeons are not available during the weekend as usually no surgeries are performed during this time.

- **Relax the instrument availability**: For each instrument set $r \in R$, q_r denotes the number of instrument sets available per day. By omitting constraint (6.4), we can determine the impact of this constraint on the number of required beds.

- **Relax all constraints**: By solving the model without all of the above mentioned constraints, we can determine the number of required beds when all resource capacities are unrestricted.

- **Relax all constraints including weekends**: Typically, no elective surgeries are performed during the weekends. However, it might be interesting to see which restriction this imposes on the objective function. Therefore, we also relaxed the availability of the ORs and surgeons during the weekends, i.e.,

the OR availability is set to five and all surgeons are available during the weekend.

	# Expected beds	# 95-perc. beds	# Times peak achieved	Int. gap (%)
Original	43.2	48	1	—
Global approach	34.7	40	14	1.06
Relax OR availability	34.4	40	5	1.93
Relax surgeon availability	34.1	40	5	1.08
Relax instrument availability	34.7	40	15	1.16
Relax all constraints	33.9	40	2	1.68
Relax all constraints including weekends	32.7	38	12	2.14

Table 6.1: Results scenarios

The results of the considered scenarios are given in Table 6.1. This table shows the expected number of required beds and the number of beds required when the 95-percentile is considered. In addition, we show the number of days in the planning horizon for which the maximum number of beds is achieved, which is denoted by '# Times peak achieved'. We also provide the integrality gap denoted by 'Int. gap'.

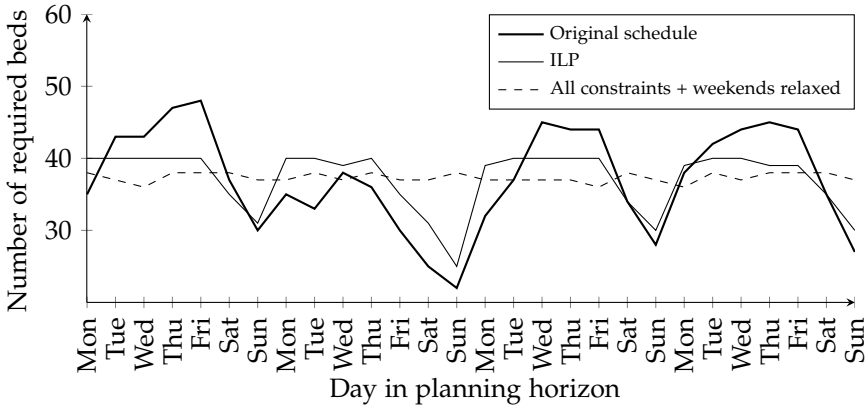


Figure 6.9: Resulting bed occupancies of three scenarios

Table 6.1 shows that the global approach reduces the number of required beds from 48 to 40 by reassigning the OR-blocks while taking into account all resource constraints. The results for the other scenarios show that the OR and surgeon availability during the week and the number of available instrument sets do not

influence the required number of beds, because for all considered scenarios, except the last one, the number of beds needed equals 40. However, the expected number of required beds and the number of times the peak bed occupancy is achieved indicate a slight improvement for the scenarios where the OR availability, the surgeon availability, or all resource constraints are relaxed. The scenario where the constraints are also relaxed during the weekends decreases the number of required beds from 40 to 38 beds. The resulting bed occupancies over the entire planning horizon for the original OR-schedule used in HagaZiekenhuis, the OR-schedule obtained by the global approach and the OR-schedule for the last mentioned scenario are given in Figure 6.9.

The differences between the bed occupancy for the original OR-schedule and the one resulting from using the global approach might be explained by the differing number of surgeries scheduled per day. For the original OR-schedule, there is a peak in the number of surgeries scheduled per day at the start of the week and halfway through the week, while for the OR-schedule created by the global approach, there is only a peak at the start of the week. The global approach also schedules OR-blocks with a high average LOS at the end of the week. Note that the bed occupancy, shown in Figure 6.9, for the global approach is rather flat during the week, however, during the weekends the bed occupancy is rather low. To flatten out these peaks, HagaZiekenhuis should consider to open the OR for elective surgeries during the weekends, because then, the number of beds needed can be reduced by two extra beds.

6.5 Conclusions and recommendations

In this chapter, we developed two approaches to improve the OR-schedule such that the number of required beds is reduced. The first approach incorporates the analytical formulation of the probability distribution of the bed occupancy and improves the OR-schedule by using a local search procedure. The second approach approximates the required number of beds by the expected bed occupancy, which enables us to solve the problem as an ILP. Both approaches are tested on 100 random instances to determine which of the two approaches provides the best solution to the original problem. The computational results show that the ILP with the simplified objective function performs the best for instances based on the situation in HagaZiekenhuis. Note that after solving the ILP, the number of required beds is still determined by using the analytic formulation. The computational results show that the number of required beds at the orthopedic department of HagaZiekenhuis can be reduced by almost 20% when the ILP is used. None of the resources used at HagaZiekenhuis restrict the improvement that can be made to the OR-schedule, however, the number of required beds can be reduced slightly when the OR is also available for elective surgeries during the weekends.

Beliën and Demeulemeester [7] considered a similar problem as discussed in this research, however, they focused on minimizing the total expected bed shortage instead of minimizing the number of required beds. They compared an SA

approach that considers the original objective function and an ILP that considers an approximation of the objective function. The approximation used in the ILP is given by the minimization of the maximum expected bed occupancy which is quite different from the original objective function that indirectly focuses on minimizing the expected bed occupancy for all days in the planning horizon. Opposite to our results, Beliën and Demeulemeester [7] conclude that the SA approach performs better than the ILP when the outcome of both approaches is compared based on the original objective function. This can be explained by the fact that the original and approximated objective function used by [7] differ significantly, while in our case, both objective functions are quite similar. Therefore, we conclude that approximating the objective function only provides good solutions to the original problem when the approximated and original objective function lead to approximately the same ordering of feasible solutions. Therefore, when using the proposed solution approach in practice, it should be verified that most feasible solutions for the considered instance are ordered in the same way by the approximated and original objective function. Further research is needed to determine which conditions of an instance lead to entirely the same ordering of feasible solutions.

The approach developed in this chapter only considers elective surgeries, because only these surgeries can be scheduled in advance. However, patients who have to undergo surgery immediately, and as a consequence, their surgery cannot be scheduled beforehand, also have to be admitted at one of the wards after surgery. By using the model of Vanberkel et al. [95], we can incorporate these emergency surgeries by introducing dummy OR-blocks that are already fixed to a specific day in the planning horizon and contain the expected number of emergency surgeries. In this way, the arrival and admission of emergency patients is considered while determining a new OR-schedule for the elective surgeries, and thus, the total number of required beds is minimized and both elective and emergency patients can be admitted after surgery. However, this approach only considers the expected number of emergency patients and does not take into account the stochastic nature of the arrival process of emergency patients. Incorporating the stochastic arrival process of emergency surgeries would be an interesting topic for further research.

The developed approach can also be used to determine the admission schedule for non-surgical patients. To achieve this, we should schedule individual admissions instead of OR-blocks. This increases the complexity of the ILP as the number of variables increases. In addition, for the case of non-surgical patients, it is not defined how many admissions can be scheduled per day as this number may be unlimited. This also increases the complexity of the ILP due to the increasing solution space. Therefore, it might be needed to improve the solution approach to guarantee a reasonable computation time.

In the considered model, we assumed that the assignment of surgery types to OR-blocks is determined beforehand by the specialism of the surgeon. However, this assignment also influences the number of required beds on the wards. Therefore, it would be interesting to also incorporate this assignment when creating an

OR-schedule such that the number of required beds can be reduced even further. Note that this also imposes some extra constraints on the model, because we also have to consider the stochastic duration of the surgeries such that the required surgical time does not exceed the available surgical time. Thus, it would be interesting to investigate this problem in future research.

Another interesting topic for future research is to take the available bed capacity at the wards into account when minimizing the number of required beds. For example, when the available bed capacity at the ward equals 40, it is not necessary to reduce the number of required beds further to 38. In addition, it might be beneficial to free as many wards as possible during the weekends to reduce the number of staff needed during the costly weekends.

Chapter 7

Clustering clinical departments for wards to achieve a prespecified blocking probability

7.1 Introduction

The HagaZiekenhuis, a hospital in the Netherlands, is forced by health insurance companies to reduce the number of beds available in the hospital with the goal to reduce the healthcare costs. Therefore, the hospital has the challenging task to treat the same number of patients with fewer beds available. As the demand for care will increase in the future due to the aging population and more chronically ill patients, the hospital expects that it even has to be able to treat more patients with this reduced bed capacity. The considered problem does not only occur in the HagaZiekenhuis, but also in other hospitals in the Netherlands, and most likely, also in hospitals in other countries.

Treating the same number of patients or even more with a reduced number of beds can be achieved in different ways. First, if the length of stay (LOS) of patients is reduced, the bed capacity needed to treat the same number of patients decreases. However, it might not always be possible to reduce the LOS, e.g., due to the specific illness of patients. A second way to reduce the bed capacity is to level the bed occupancy over time. This can, for example, be done by adjusting the operating room schedule as in Chapter 6. Third, by clustering several (small) clinical departments (i.e. by assigning several clinical departments to the same wards such that the departments share the beds available at these wards), the bed capacity can also be reduced. The basic reason for this is that when two or more clinical departments are clustered, also the risk of refusing a patient is reduced which leads to less beds needed in total.

In this chapter, we consider the third option to solve the problem of HagaZiekenhuis. Thus, we aim to assign one or more clinical departments to a cluster such that the number of beds needed in total is reduced. While determining the clusters, we consider the medical limitations meaning that, for example, the general surgery and internal medicine department cannot be clustered. Each formed cluster is then assigned to one or more wards such that enough beds are available on these assigned wards to guarantee that the probability of refusing a patient lies below a given threshold. Note that not more than one cluster can be assigned to a ward to ensure that only the clinical departments assigned to the considered cluster share the beds available at these wards. We do not consider, for example, the varying bed

occupancy during the week and weekends and the arrangement of rooms on the wards as we only consider the strategic problem. These more detailed decisions can be made at a later stage when the assignment of clinical departments to wards is fixed.

Several papers discuss methods to determine the number of beds needed for a specific clinical department such that almost all patients can be admitted. Lapierre et al. [60] consider the problem of assigning a number of beds to clinical departments while taking into account that the admission schedule varies over the week and that the demand can show seasonality. They develop a time series model that predicts when the demand exceeds the available number of beds. This model can help in making good decisions regarding the size of a ward.

Green and Nguyen [44] use an $M/M/s$ queuing model to estimate the delay of admission when a number s of beds is given. By considering several values for s, they can chose the number of beds such that the resulting delay of admission is acceptable for hospital management. Harper and Shahani [48] present a detailed simulation model which determines the refusal rate for a given bed capacity and planning policy.

Gorunescu et al. [42] use an Erlang loss model ([88]) to determine the number of beds needed by a clinical department. Hereby, they make a trade-off between the loss probability and the costs of empty beds. Gorunescu et al. [43] extend this approach by introducing a waiting room with extra beds and show that this waiting room improves the performance of the system.

Utley et al. [90] use a generating function to represent the probability distribution of the bed occupancy for each day of a given planning horizon. This probability distribution includes the scheduled arrival of elective patients while considering a no-show probability and the unscheduled arrival of emergency patients. An unlimited bed capacity is assumed such that the probability distribution represents the bed occupancy when none of the patients is blocked. With the use of this probability distribution, the number of beds needed and the corresponding blocking probability can be determined.

Kokangul [57] models the bed occupancy as the difference between two renewal processes, namely the arrival and departure process. A simulation of the bed occupancy process is used to investigate the relationship between the optimal number of beds and the bed occupancy, number of admissions, and service level. The results of this simulation are used to determine mathematical relationships between these factors. These relationships are then incorporated in a nonlinear mathematical model which is used to determine an optimal number of beds for each clinical department.

Li et al. [64] developed a goal programming approach to make a trade-off between the number of beds required to achieve a targeted acceptance probability and the number of beds needed to optimize daily profits. The number of beds needed is determined by means of the Erlang loss formula.

De Bruin et al. [26] developed a decision support system based on the Erlang loss model to determine the appropriate size of wards. De Bruin et al. [26] and

Green and Nguyen [44] also discuss the benefits of merging departments. Green and Nguyen [44] show that consolidating clinical departments reduces the number of beds needed and increases the total bed occupancy rate, especially when several small clinical departments are clustered. De Bruin et al. [26] show that merging clinical departments results in an improved blocking probability and bed occupancy rate as larger departments are formed.

As in several of the mentioned papers, we also use the Erlang loss formula to determine the number of beds needed for each cluster of clinical departments while considering the blocking probability chosen by the hospital. We incorporate the Erlang loss formula into a model that determines which clinical departments should be clustered and that also assigns the clusters to one or more wards. The assignment of clusters to wards is done in such a way that enough beds are available at the assigned wards to guarantee a given blocking probability. While defining the clusters, we consider the medical limitations as mentioned before, but we also try to limit the number of clinical departments assigned to one cluster. When assigning the clusters to wards, we consider the preference of clinical departments for a specific ward and the distance between the wards assigned to a specific cluster. More detailed information about the chosen modeling approach can be found in Section 7.2. In Section 7.3, an exact solution approach for this problem is discussed. As within the practical setting of HagaZiekenhuis not always precise knowledge is available on the input parameters needed for the model or these parameters might change over time, we have to be able to evaluate several what-if scenarios in order to present a robust solution. Furthermore, since the computation time for the exact solution approach is quite long, we introduce two heuristic solution methods to reduce the computation time. The first heuristic solution approach approximates the Erlang loss formula leading to a simpler model needing less computational time. The second heuristic solution approach solves the problem in two phases. The first phase clusters the clinical departments and the second phase assigns the clusters to wards. By splitting the problem up in two phases, the overall computation time is reduced. In Section 7.4, all three solution methods are tested on instances based on the situation at HagaZiekenhuis and compared according to the computation time and solution quality. In addition, we demonstrate the potential of the solution methods to investigate several scenarios and to evaluate the robustness of a given solution. Section 7.5 presents conclusions and gives recommendations for further research.

Summarizing, the contribution of this chapter is as follows. First, we introduce a problem, which in this combination has not been considered before in literature. Many papers already discussed clustering clinical departments and using the Erlang loss model to determine the number of beds needed, however, none of the mentioned papers investigate which clinical departments should be clustered such that all departments have enough beds available on the assigned wards. Second, we provide an exact solution method and two heuristic solution methods. The latter are needed to quickly solve this problem such that several scenarios can be evaluated.

7.2 Problem formulation

Each hospital consists of several clinical departments, such as general surgery, neurosurgery, orthopedics, internal medicine, and cardiology. These clinical departments need a certain number of beds to admit and treat all their patients. The needed number of beds depends on the average LOS μ of patients and the expected number of admissions per day λ. The values for μ and λ are in general different for each clinical department. Some departments such as general surgery and orthopedics admit a high number of patients per day which each have a relative short LOS (e.g., three days). The average LOS for internal medicine is often much higher (approximately six days). If the average LOS μ and the number of admissions λ would be deterministic, i.e., each day exactly λ patients arrive with LOS μ, then the corresponding clinical department needs $\lambda\mu$ beds to be able to admit all patients. However, in practice μ and λ are stochastic, since the LOS varies per patient and the number of admissions varies per day for each clinical department. To make sure that enough beds are available in a stochastic setting, we surely need more beds than $\lambda\mu$ to deal with the fluctuating number of admitted patients. However, in practice, it will not be possible to always admit all patients, because then the number of beds needed has to be extremely large and a large portion of the beds will be empty most of the time, and thus, the available capacity is not used efficiently. Therefore, the goal is to be able to admit the patients with a high probability.

Using the Erlang loss model, we can determine the probability of not being able to admit a patient, also known as the blocking probability, when μ, λ, and the number of available beds x are given. The resulting blocking probability is given by (see e.g. [88]):

$$P_{\text{loss}}(\lambda\mu, x) = \frac{(\lambda\mu)^x / x!}{\sum_{k=0}^{x} (\lambda\mu)^k / k!}. \tag{7.1}$$

However, for the problem considered in this chapter, we would like to determine the number of beds needed when μ, λ, and the blocking probability B are given. The values for μ and λ can be determined from historical data provided by the hospital and the hospital itself can chose an acceptable value for the blocking probability B. When μ, λ, and B are given, the goal is to find the smallest number of beds x for which $P_{\text{loss}}(\lambda\mu, x)$ is smaller than or equal to B. This can either be done by complete enumeration or for example by the bisection method of Qiao and Qiao [79].

From equation (7.1) it also can be derived that clustering clinical departments can reduce the number of beds needed in total. Consider for example two clinical departments with $\mu_1 = 7.1$, $\lambda_1 = 9$ and $\mu_2 = 6.6$, $\lambda_2 = 3$ respectively. When both departments have their own ward and the blocking probability is set to 0.05, then the first department would need 70 beds and the second department would need 25 beds, which leads to 95 beds in total. But when both departments share a ward, μ and λ become $\frac{\mu_1\lambda_1+\mu_2\lambda_2}{\lambda_1+\lambda_2} = 6.975$ and $\lambda_1 + \lambda_2 = 12$ respectively for the formed

cluster, which leads to 89 beds in total. Thus, for this small example the number of beds can be reduced by six by clustering both departments. For the problem considered in this chapter the number of beds available at the wards is limited, and thus, clustering departments might be needed to ensure that the number of beds assigned to a cluster is enough to guarantee a given blocking probability.

The problem considered in this chapter can be defined as follows. Let D be the set of clinical departments and let W be the set of available wards. For each clinical department $d \in D$ the average LOS is given by μ_d and the expected number of admissions per day by λ_d. The number of available beds at ward $w \in W$ is given by b_w and is assumed to be fixed. The clinical departments must be assigned to the wards such that enough beds are available at the wards to admit patients of the assigned clinical department with a probability $1 - B$. Because in practice not all clinical departments can have their own ward, we first form clusters that consist of one or more clinical departments. Note that each clinical department is assigned to exactly one cluster such that the clusters form a partition of the clinical departments (see Figure 7.1). Next, the clusters are assigned to one or more wards such that the resulting blocking probability is below B for each cluster. Note that not more than one cluster can be assigned to a ward to ensure that the beds on this ward are only shared by the clinical departments assigned to this cluster (see Figure 7.1). Clustering departments is not for free, since it may complicate the process of doing rounds as the patients of one clinical department might be scattered over multiple wards. Therefore, we have to ensure that the clinical departments are clustered as less as possible.

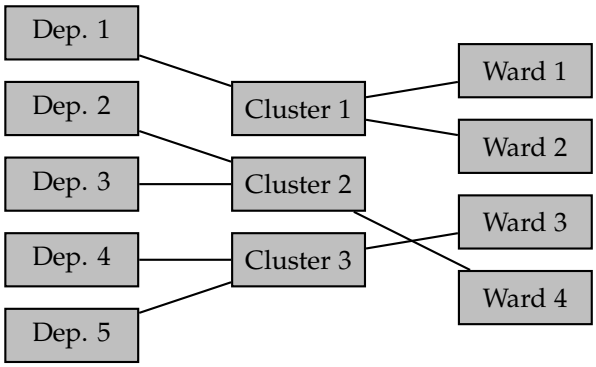


Figure 7.1: Example of a feasible assignment

The assignment of clusters to wards can be considered as a bin packing problem where the bins have varying sizes. Therefore, the number of beds available on the assigned wards will in most cases be larger than the number of beds needed by the assigned cluster which can be considered as slack.

To create a feasible solution, each clinical department $d \in D$ has to be assigned to exactly one cluster $c \in C$, where C is the set of clusters. Note that the number of clusters is not prespecified, and thus, the cardinality of C is part of the decision

process. To incorporate this we choose C to be sufficiently large, i.e., equal to $\max\{|D|, |W|\}$, but allow some clusters $c \in C$ to be empty after the assignment process. To model the assignment of clinical departments to clusters, we introduce binary variables X_{dc} which are one when clinical department $d \in D$ is assigned to cluster $c \in C$ and zero otherwise. To ensure that each clinical department $d \in D$ is assigned to exactly one cluster $c \in C$, we introduce the following constraint:

$$\sum_{c \in C} X_{dc} = 1, \qquad \forall d \in D. \tag{7.2}$$

In practice, not all clinical departments are allowed to share a ward due to medical reasons. For example, patients who had surgery, and thus, have a wound that needs to heal, cannot be placed at the same ward as patients who have an infectious disease. These limitations are specified by the hospital and are modeled by parameters a_{de} which are one when clinical department $d \in D$ is allowed to be in the same cluster as clinical department $e \in D$ and zero otherwise. To ensure that clinical departments are only clustered when this is allowed, the following constraint is imposed:

$$X_{dc} + X_{ec} - 1 \leq a_{de}, \quad \forall c \in C, d, e \in D. \tag{7.3}$$

The generated clusters must be assigned to one or more wards such that enough beds are available for the clusters. For this, binary variables Y_{cw} are introduced, which are one when cluster $c \in C$ is assigned to ward $w \in W$ and zero otherwise. As stated before, at most one cluster may be assigned to a ward, which is ensured by the following constraint:

$$\sum_{c \in C} Y_{cw} \leq 1, \qquad \forall w \in W. \tag{7.4}$$

The number of required beds V_c for a cluster $c \in C$ can be determined by the Erlang loss formula (equation (7.1)). Note that this number V_c cannot be determined straightforward, and thus, cannot be expressed in linear constraints. Assuming this number to be known, we have to ensure that the total number of beds available at the assigned wards is greater than or equal to V_c. This is ensured by the following constraint:

$$V_c \leq \sum_{w \in W} Y_{cw} b_w, \qquad \forall c \in C. \tag{7.5}$$

The quality of a solution for HagaZiekenhuis, and also other hospitals, depends on several aspects. Therefore, the chosen objective function consists of three parts. First, we do not want to cluster more clinical departments into one cluster than necessary to keep the processes at the wards simple and clear. Therefore, we aim

to minimize the maximum number of clinical departments in one cluster. The second part of our objective function is to assign clusters to wards that are close to each other as this is one of the requirements given by HagaZiekenhuis. For this, we denote for each pair of wards $w, v \in W$ the distance between those two wards by g_{wv} and add a term to the objective function that minimizes the sum of all pairwise distances of wards assigned to the same cluster. Third, some clinical departments in HagaZiekenhuis have a preference for a certain ward because of its location. For example some clinical departments, such as general surgery, orthopedics, and neurosurgery, want to be close to the operating theater. Other clinical departments, such as cardiology and thoracic surgery, want to be close to the intensive care unit. These preferences of clinical department $d \in D$ for wards $w \in W$ are denoted by h_{dw}. Combining these three objectives, the resulting objective function is:

$$\min \left(\max_{c \in C} \sum_{d \in D} X_{dc} + \sum_{c \in C} \sum_{w \in W} \sum_{v \in W} g_{wv} Y_{cw} Y_{cv} - \sum_{c \in C} \sum_{d \in D} \sum_{w \in W} h_{dw} X_{dc} Y_{cw} \right). \quad (7.6)$$

Note that the problem can easily be divided into two phases. In the first phase, clinical departments are clustered, and in the second phase, the clusters are assigned to wards. However, even when we assume that the assignment of clusters to wards is already fixed, the resulting problem is strongly $\mathcal{NP}$-hard. To see this, assume that there are t clusters and for each cluster b beds are available on the assigned wards. Next, assume $3t$ clinical departments, where clinical department $d \in D$ needs a_d beds and $\sum_{d=1}^{3t} a_d = tb$. For this instance, determining a feasible assignment of clinical departments to wards is equivalent to determining whether there are t disjoint subsets $R_l \subset \{1, \ldots, 3t\}$ such that $\sum_{d \in R_l} a_d = b$ for $l = 1, \ldots, t$, which is known as the 3-partition problem [36]. With a similar argument we can prove that the problem is also strongly $\mathcal{NP}$-hard when the assignment of clinical departments to clusters is fixed. This means that the problem consists of a combination of two strongly $\mathcal{NP}$-hard problems.

7.3 Solution methods

For the presented problem, the average LOS μ and the expected number of admissions per day λ are important input parameters. However, these parameters are both very uncertain and may change over the years: the average LOS is likely to reduce as surgeries become less invasive and diagnoses are made quicker, and due to the aging population the expected number of admissions per day is likely to increase as more and more people need care. This has as a consequence that in practice, several what-if scenarios have to be evaluated, and thus, fast solution methods for the stated problem are required. In Section 7.3.2, we propose such fast heuristic solution methods. Before that, in Section 7.3.1, we discuss an exact solution method. This method is used to determine optimal solutions, which are used to evaluate the performance of the heuristic solution methods.

7.3.1 Exact solution method

The basis of the exact solution method is the formulation given in the previous section. This formulation is used to set up an Integer Linear Program (ILP) to solve the problem. The main difficulty of formulating the given problem as an ILP is determining V_c, i.e., the number of beds needed for a certain cluster $c \in C$. As we are not aware of an exact method to determine V_c by means of linear equations, we need to derive another method to incorporate this element. Note that when the blocking probability B is given, the number of beds needed for a certain cluster is completely determined by $\lambda\mu$, which represents the expected number of admitted patients per day. Therefore, we first determine the number of beds needed for each possible value of $\lambda\mu$ for the given problem. As the number of beds needed can only be integer, the resulting function $f_B(\lambda\mu)$ is a stepwise function. In Figure 7.2, an example is shown with B equal to 0.05 and 0.1.

From the two functions in Figure 7.2 it can be seen that more beds are needed when the blocking probability decreases. This should be the case, since when the blocking probability decreases, we need a larger buffer of beds to guarantee this blocking probability. When solving the problem in practice, the value of B is chosen in advance by the hospital. Therefore, the function $f_B(\lambda\mu)$ to be used can be predetermined. It remains to determine the number of beds V_c needed for each cluster $c \in C$. To determine this number, we first have to determine the product of the values λ and μ for each cluster $c \in C$. This value $(\lambda\mu)_c$ is calculated by:

$$(\lambda\mu)_c = \sum_{d \in D} X_{dc}\lambda_d\mu_d, \quad \forall c \in C. \tag{7.7}$$

Then, the number of beds V_c needed for cluster $c \in C$ can be determined as follows:

$$V_c = f_B\left((\lambda\mu)_c\right), \qquad \forall c \in C. \tag{7.8}$$

As the function $f_B(\lambda\mu)$ used in constraint (7.8) is not linear but a stepwise function, we have to use standard integer linear programming techniques to incorporate constraints (7.7) and (7.8) into the ILP (see e.g. [96]). For each cluster $c \in C$, we need to the determine the interval of the stepwise function $f_B(\lambda\mu)$ in which $(\lambda\mu)_c$ lies. This leads to an extra binary variable for each cluster and interval pair. To determine this interval, we need two constraints for each cluster. The first constraint, which combines constraints (7.7) and (7.8), determines the correct interval of the stepwise function $f_B(\lambda\mu)$. The second constraint ensures that only one interval is selected. Furthermore, we need a constraint similar to constraint (7.5) to ensure that enough beds are available on the wards assigned to cluster $c \in C$ to guarantee the loss probability B.

Another difficulty of the formulation described in Section 7.2 is the quadratic part of the objective function (7.6). However, this easily can be linearized by introducing binary variables Z_{dcw} and T_{cvw}. Binary variables Z_{dcw} are set to one when

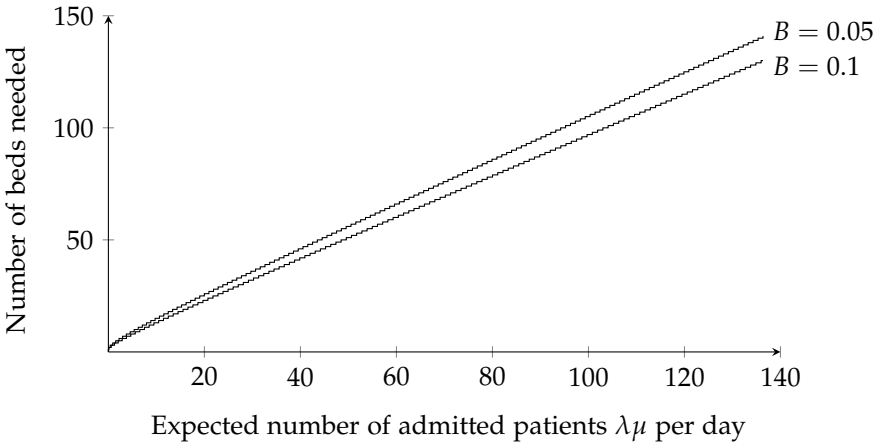


Figure 7.2: Exact stepwise function

department $d \in D$ is assigned to cluster $c \in C$ which on its part is assigned to ward $w \in W$ and zero otherwise. The correct values of Z_{dcw} are determined by standard integer linear programming tricks which lead to the following three constraints:

$$\begin{aligned} &Z_{dcw} \leq X_{dc}, && \forall d \in D, c \in C, w \in W, \\ &Z_{dcw} \leq Y_{cw}, && \forall d \in D, c \in C, w \in W, \\ &Z_{dcw} \geq X_{dc} + Y_{cw} - 1, && \forall d \in D, c \in C, w \in W. \end{aligned} \tag{7.9}$$

We set binary variables T_{cvw} equal to one when both ward $v \in W$ and $w \in W$ are assigned to cluster $c \in C$ and zero otherwise. The correct value of T_{cvw} is determined by constraints similar to constraints (7.9).

Except from the quadratic part, the objective function (7.6) also contains a min-max term. The maximum of $\sum_{d \in D} X_{dc}$ over c can easily be determined by adding an extra constraint to the formulation.

Concluding, the described problem can be solved to optimality by solving an integer linear program. However, the solving time is large since based on the above described modeling techniques the resulting instances are already quite large for moderate input instances. Therefore, we discuss several fast heuristic solution methods in the next section.

7.3.2 Heuristic solution methods

To be able to solve the considered problem in reasonable computation time, we consider two heuristic solution methods. The first method approximates the stepwise function $f_B(\lambda\mu)$ which might reduce the computation time as the number of integer variables reduces. The second method solves the problem in two phases. The first phase clusters the clinical departments and the second phase assigns the

clusters to wards. By splitting up the problem, we hope to reduce the computation time.

Approximation solution method

Figure 7.2 already indicates that the stepwise function $f_B(\lambda\mu)$ is 'close' to linear in the considered interval except at the origin. This indicates that it might be possible to approximate the stepwise function by a linear function to reduce the computation time. To further investigate the linear behavior of the stepwise function, we consider the step sizes, i.e., the lengths of the intervals for which the number of beds needed is constant, for both evaluated values of B, which are shown in Figure 7.3.

As the slope of the stepwise function is higher for $B = 0.05$ than for $B = 0.1$, it is no surprise that the step sizes are larger for $B = 0.1$ when compared to $B = 0.05$. In addition, Figure 7.3 shows that the step sizes converge rather quickly for both values of B. When the number of beds is 20 or more, the step size is almost constant which indicates an almost constant slope. As most wards consist of more than 20 beds, we may easily approximate the stepwise function by a linear function.

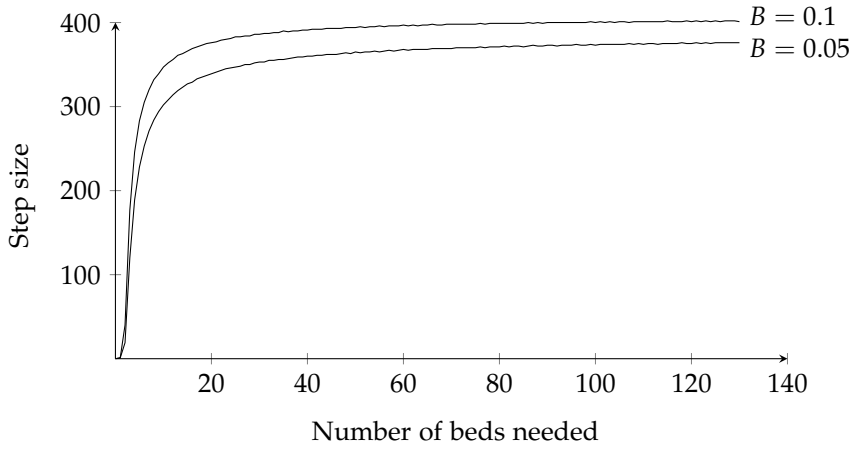


Figure 7.3: Step sizes of stepwise function

Shakhov [82] proved that the number of beds x needed can be approximated by $\lambda\mu(1 - B)$. However, this approximation only holds when $\lambda\mu$ is sufficiently large. As we only want to approximate x for the interval considered in Figure 7.2, we introduce the following linear approximation.

Because we want to guarantee the blocking probability B, we do not want to underestimate the number of beds needed. Therefore, we determine a linear function which never underestimates the stepwise function in the considered interval when rounded up to the nearest integer. In addition, we minimize the maximum overestimation of the stepwise function in the considered interval, because this improves

the quality of the solution when compared to the optimal solution. When the overestimation is too large, it may happen that a certain cluster cannot be assigned to a ward, because the number of available beds seems to be too small. However, when the number of needed beds is determined exactly, i.e. by the stepwise function, the number of available beds might be sufficient. Therefore, we minimize the maximum overestimation to approximate the optimal solution as good as possible.

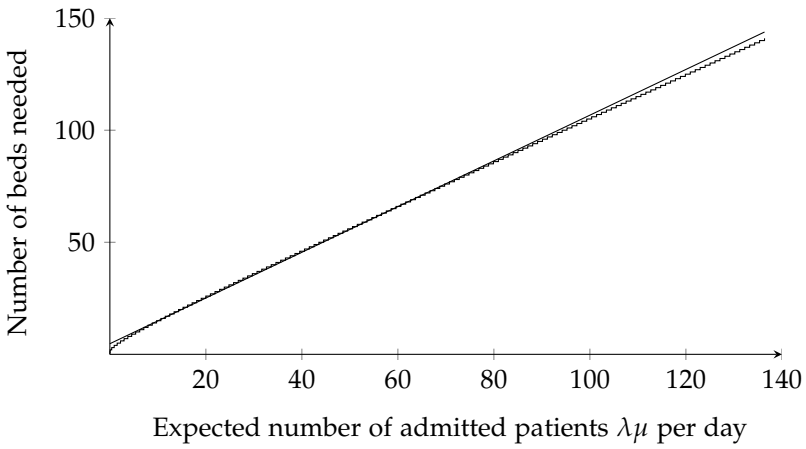


Figure 7.4: Stepwise function and approximation stepwise function for $B = 0.05$

We use the following mixed integer program (MIP) to determine the linear function which satisfies the two stated criteria, where P and R are the coefficients of the linear function used to approximate the stepwise function. Note that to linearize the given constraint and objective function, we need additional integer variables.

$$\begin{aligned} \min \quad & \max_{\lambda\mu} \lceil P(\lambda\mu) + R \rceil - f_B(\lambda\mu) \\ \text{s. t.} \quad & \lceil P(\lambda\mu) + R \rceil \geq f_B(\lambda\mu), \qquad \forall \lambda\mu, \\ & P, R \in \mathbb{R}. \end{aligned} \tag{7.10}$$

Figure 7.4 shows the stepwise function and linear approximation for $B = 0.05$. For the considered interval, the maximum overestimation equals four beds, which is achieved at the beginning and end of the interval, i.e., when less than 0.05 or more than 120 patients are admitted per day. Note that in practice the number of beds needed by a cluster often lies between 20 and 60. As Figure 7.4 shows, the overestimation is much smaller in this region, i.e., at most one bed.

Using the above linear approximation, we simply add the following constraint to the formulation given in Section 7.2 to determine the number of required beds:

$$V_c = P(\lambda\mu)_c + R, \qquad \forall c \in C. \tag{7.11}$$

Note that we do not need to round the number up as the number of beds available on the assigned wards should be larger than or equal to the number of beds needed by the cluster.

By using this linear approximation function, we can eliminate the binary variables and the two constraints needed to determine the right interval of the stepwise function. However, to ensure that $V_c = 0$ when cluster $c \in C$ is empty, we do have to add binary variables which denote whether cluster $c \in C$ is empty or not. To determine the correct value of these binary variables, we need two constraints for each cluster $c \in C$. This means that the number of constraints stays the same, but instead of a binary variable for each cluster and interval pair, we only need a binary variable for each cluster. Thus, the number of binary variables reduces significantly when an approximation of the stepwise function is used.

Hybrid heuristic

As already mentioned in Section 7.2, the considered problem can be divided into two phases. In the first phase, clinical departments are clustered, and in the second phase, the clusters are assigned to wards. This approach has some similarities with column generation [24], which is an iterative process of generating columns for a linear program and selecting a subset of these columns. However, in our case, we do not only need to select a subset of clusters such that every clinical department is chosen once, but we also need to assign the clusters to one or more wards to guarantee a feasible solution. Therefore, the column generation approach in its original form is not suitable for the problem considered in this chapter. However, the basic idea of solving the problem in two phases is an interesting approach.

In the column generation approach, new columns are generated on the fly. Applying this to our problem would mean that each step in the iterative process consists of two phases, where in the first phase new clusters are added to the already generated set of clusters, and in the second phase, the new complete set of clusters is used to select a subset of these clusters. We modify this approach as we completely separate the first and second phase, i.e., we do not perform both phases in an iterative manner. We first execute the first phase by generating a large set of feasible clusters. In the second phase, we select a subset of these clusters which forms a partition of the clinical departments and assign them to wards. As the set of clusters generated in the first phase is too large to completely be considered in an ILP approach for the second phase, we perform a local search method which only considers a subset of this large set in each step. Note that during the local search procedure, the large set is not changed. Only the subset considered in each step of the local search procedure changes.

The first phase in the two phase approach is generating clusters or in other words assigning clinical departments to clusters. The only constraints considered

when forming the clusters in the first phase are the medical limitations and that there must be at least one subset of clusters that forms a partition of the clinical departments. When the complete generation process is done randomly, there is a risk that no subset of clusters exists which form a partition of the clinical departments, and thus, no feasible solution will exist. Therefore, we first generate two sets of clusters which guarantee that such a partition exists. The first set of clusters represents a partition of the clinical departments for which a small number of beds is needed. This partition should ensure that with high probability a feasible assignment of clusters to wards exists. Note that in general a small number of clusters leads also to a small number of beds needed. Therefore, we search for a clustering with a minimal number of clusters. This set of clusters can easily be generated by solving the following ILP for $|C| = 1$, $|C| = 2$, etc. until a feasible solution is found.

$$\begin{array}{ll} \sum_{c \in C} X_{dc} = 1, & \forall d \in D, \\ X_{dc} + X_{ec} - 1 \leq a_{de}, & \forall c \in C, d, e \in D, \\ X_{dc} \in \{0,1\}, & \forall d \in D, c \in C. \end{array} \tag{7.12}$$

We also introduce a set of clusters, where each clinical department forms a cluster on its own. These clusters might be useful in finding feasible partitions of the clinical departments when the clusters are assigned to the wards.

As the clusters described above most likely will not lead to an optimal solution, we also generate random clusters. For each of these N clusters, we start with selecting one of the clinical departments at random. Next, we determine the set $\bar{D}$ of clinical departments that are allowed to share a ward with the selected clinical department, and thus, are allowed to be assigned to the same cluster. To determine the (maximum) size of the generated cluster, i.e., the number of clinical departments allowed to be assigned to this cluster, we randomly determine a number between one and the cardinality of $\bar{D}$. Then, we randomly add clinical departments from $\bar{D}$ to the cluster until the maximum size is achieved or until none of the not yet assigned clinical departments from $\bar{D}$ can be added due to the medical limitations. When a generated cluster is the same as one of the previously generated clusters, we start the process again to ensure that all N generated clusters are unique. The process of generating the N random clusters is explained in more detail by the following algorithm.

Repeat until N clusters are added to set C:

Step 1. Open a new cluster $\bar{c}$.

Step 2. Select a clinical department $d \in D$ at random and assign it to cluster $\bar{c} \in C$, i.e., $X_{d\bar{c}} = 1$.

Step 3. Determine set $\bar{D} \subseteq D$ of clinical departments which are allowed to be assigned to the same cluster as the selected clinical department d. Note that $d \notin \bar{D}$.

Step 4. Randomly draw a number from $\{2, \ldots, |\bar{D}|\}$ to determine the (maximum) size S of cluster $\bar{c} \in C$. Note that we do not generate clusters of maximum size or of size one as these these clusters are already present in the two extreme solutions.

Step 5. Repeat until $\sum_{d \in D} X_{d\bar{c}} = S$ or $\bar{D} = \emptyset$:

- Step (a) Select a clinical department $d \in \bar{D}$ at random and assign it to cluster $\bar{c} \in C$, i.e., $X_{d\bar{c}} = 1$.
- Step (b) Reduce $\bar{D}$ to the set of clinical departments which are allowed to be assigned together with all clinical departments in $\bar{c}$.

Step 6. If $\bar{c} \notin C$, add $\bar{c}$ to C, i.e., $C := C \cup \{\bar{c}\}$.

After this phase 1, we need to determine which combination of clusters leads to the best solution. As the chosen number N of generated clusters is generally quite large to guarantee a good feasible solution, selecting the best clusters from this set with the use of an ILP would take too long. Therefore, we have developed a local search procedure which solves a smaller ILP in each iteration. More precisely, in each iteration of the local search approach, we select a subset of the generated clusters $\bar{C} \subset C$ and use an ILP to determine which set of the selected clusters leads to the best solution when considering only the clusters in $\bar{C}$. The ILP used for this step is a reduced version of the ILP formulation introduced in Section 7.3.1. To determine whether a cluster is used in the optimal solution of this ILP, we need extra binary variables Γ_c which are one when cluster $c \in \bar{C}$ is selected and zero otherwise. Note that as the clusters are generated before solving this ILP, variables X_{dc} are known and fixed, and the number of beds V_c needed by cluster $c \in \bar{C}$ can be determined beforehand. This means that the only free variables are the binary variables Y_{cw} and Γ_c.

$$\min \left(\max_{c \in \bar{C}} \sum_{d \in D} X_{dc}\Gamma_c + \sum_{c \in \bar{C}} \sum_{w \in W} \sum_{v \in W} g_{wv} Y_{cw} Y_{cv} - \sum_{c \in \bar{C}} \sum_{d \in D} \sum_{w \in W} h_{dw} X_{dc} Y_{cw} \right) \tag{7.13}$$

$$\text{s. t.} \quad \sum_{c \in \bar{C}} X_{dc}\Gamma_c = 1, \qquad \forall d \in D, \tag{7.14}$$

$$Y_{cw} \leq \Gamma_c, \qquad \forall c \in \bar{C},\ w \in W, \tag{7.15}$$

$$\sum_{c \in \bar{C}} Y_{cw} \leq 1, \qquad \forall w \in W, \tag{7.16}$$

$$V_c\Gamma_c \leq \sum_{w \in W} Y_{cw} b_w, \qquad \forall c \in \bar{C}. \tag{7.17}$$

$$Y_{cw}, \Gamma_c \in \{0,1\} \qquad \forall c \in \bar{C},\ w \in W \tag{7.18}$$

The objective function of the ILP is the same as (7.6) except that it is restricted to the selected clusters instead of all clusters in the chosen subset $\bar{C}$. Constraint (7.14)

is introduced to select a set of clusters such that each clinical department $d \in D$ is selected exactly once. Constraint (7.15) ensures that only the selected clusters are assigned to one or more wards. Constraint (7.16) is exactly the same as constraint (7.4) and constraint (7.17) differs slightly from constraint (7.5) to only consider the beds needed by the selected clusters.

To find a good solution to the original problem, we use a local search approach. A local search approach starts with an initial solution and then iteratively moves to a neighbor solution. Hereby, a solution is given by a set $\bar{C} \subset C$ of Q clusters and the objective value of this solution is given by the optimal solution of the ILP given by (7.13)-(7.17). As starting solution, we select a subset of clusters $\bar{C}$ which includes the set of clusters for which the clinical departments are partitioned in a minimal number of clusters and the set of clusters where each clinical department has its own cluster. The subset of clusters $\bar{C}$ for the initial solution is further completed with randomly chosen clusters from C. For this initial subset of clusters, the ILP given by (7.13)-(7.17) is solved. Note that the described initial solution does not always guarantee that a feasible solution to the ILP exists. However, including the two mentioned sets of clusters increases the probability of a feasible solution existing. When no feasible solution exists, the set of randomly selected clusters is replaced by a new set of randomly selected clusters until a feasible initial solution is found. The neighborhood of a solution $\bar{C}$ is defined by all solutions $\tilde{C} \subset C$ which contain the clusters selected in the optimal solution of the ILP belonging to $\bar{C}$, i.e., clusters $c \in \bar{C}$ for which $\Gamma_c = 1$. By randomly adding clusters from C to the subset of clusters forming the optimal solution of $\bar{C}$, we randomly choose one of the neighbor solutions. The following algorithm further describes the local search approach, where Q denotes the cardinality of the chosen set $\bar{C}$, which is the same for all iterations.

Step 1. Select a subset of the generated clusters $\bar{C} \subset C$ with $|\bar{C}| = Q$ that includes the two starting solutions mentioned before.

Step 2. Solve the ILP given by (7.13)-(7.17) and let $\bar{L}$ denote the objective function value of the corresponding optimal solution.

Step 3. Repeat until K consecutive iterations are executed without reducing $\bar{L}$:

- Step (a) Generate a new subset of clusters $\bar{C} \subset C$ with $|\bar{C}| = Q$ that includes the clusters selected in the optimal solution of the ILP in the previous iteration.
- Step (b) Solve the ILP given by (7.13)-(7.17) and let L denote the objective function value of the optimal solution.
- Step (c) If $L < \bar{L}$ set $\bar{L} := L$.

Note since the clusters selected in the previous optimal solution are again in $\bar{C}$, we always have a feasible solution for the ILP and the objective function value L in step 3b is less than or equal to the objective function value of the previous iteration. Therefore, the sequence of the $\bar{L}$-values is non-increasing.

7.4 Computational results

This section aims to compare the three described solution methods with respect to the computation time and the quality of the solutions based on data provided by HagaZiekenhuis, which initiated this research. In addition, we want to show the usefulness and applicability of the methods when used in practice.

The used data consists of 16 clinical departments and 13 wards for which the number of beds varies from 8 to 32. The lay-out of the hospital is depicted in Figure 7.5, where the elevator and stairs are positioned in between side A and B. The number of beds available on each ward is depicted between brackets. Note that 16 wards are shown, however, wards 9A, 8A, and 8B are not equipped to admit patients. When two wards are on the same floor, the distance between these two wards is set zero. When two wards are on different floors but on the same side of the building, i.e., both are on side A or B, the distance is the distance in floors. When the two wards are on opposite sides of the building and on different floors, the distance equals the distance in floors plus one.

11A (28)	11B (32)
10A (32)	10B (32)
~~9A~~	9B (32)
~~8A~~	~~8B~~
7A (32)	7B (32)
6A (32)	6B (32)
5A (32)	5B (30)
4A (8)	4B (24)

Figure 7.5: Lay-out wards and number of beds available

The terms of the objective function might be conflicting, e.g., more clinical departments in one cluster may reduce the sum of the distances between clusters. However, in practice, we want to make a fair trade-off between these conflicting objectives. Therefore, we multiply the first term by ten such that this part of the objective has the same value range as the other two terms.

The medical restrictions are that all surgical departments can be clustered and all non-surgical departments can be clustered. Furthermore, the cardiology department cannot be in the same cluster as any of the other departments as the cardiology department is quite large. Therefore, the number of clusters in the optimal solution for our problem instance is at least three. When the blocking probability is set to 0.05, the total number of beds needed for this solution with only three clusters is 321, which therefore is a lower bound on the number of beds needed. When the clinical departments are not clustered at all, we obtain an upper bound on the number of beds needed which equals 365. The total number of beds available on the wards equals 378. This means that in theory, the number of beds is sufficient

to accommodate all patients when none of the clinical departments are clustered. However, this does not guarantee that we have a feasible solution as we still have to find a feasible clustering and assignment of clusters to wards.

During testing, we realized that the given ILP formulations lead to a large number of symmetric solutions and that this leads to large computation times for solving the ILP. Therefore, we first discuss the use of symmetry-breaking constraints in Section 7.4.1. In Section 7.4.2, we report on experiments with different values for the parameters used in the hybrid heuristic to determine which values result in the best performance. In Section 7.4.3, we discuss the computational results of the considered solution methods with varying blocking probabilities. In Section 7.4.4, we investigate the effect of changes in the uncertain parameters μ and λ by considering several scenarios.

All three solution methods are implemented in AIMMS 3.11 and run on an Intel Core2 Duo CPU P8600 2.40 GHz with 3.45 GB RAM. The ILPs are solved with CPLEX 12.3.

7.4.1 Symmetry-breaking constraints

The ILP formulated in Section 7.3.1 is completely symmetric with respect to the clusters. For any solution, an equivalent solution can be constructed by switching the clinical departments assigned to any pair of clusters. This results in $|C|!$ equivalent solutions based on the permutation of clusters. The number of symmetric solutions can be reduced if we order the clusters with respect to one of their characteristics. We consider the following three characteristics: (1) the number of clinical departments assigned to a cluster, (2) the value of $(\lambda\mu)_c$ which represents the expected number of patients treated on a given day, and (3) the number of beds needed for a cluster to guarantee the predefined blocking probability. The ordering with respect to these three characteristics can be formulated by the following constraints:

$$\sum_{d \in D} X_{dc} \geq \sum_{d \in D} X_{d,c-1}, \qquad \forall c \in C \text{ and } c > 1, \tag{7.19}$$

$$\sum_{d \in D} X_{dc}\lambda_d\mu_d \geq \sum_{d \in D} X_{d,c-1}\lambda_d\mu_d, \qquad \forall c \in C \text{ and } c > 1, \tag{7.20}$$

$$V_c \geq V_{c-1}, \qquad \forall c \in C \text{ and } c > 1. \tag{7.21}$$

Constraint (7.19) represents the ordering according to the number of clinical departments assigned to a cluster, constraint (7.20) the ordering according to the expected number of patients treated per day, and constraint (7.21) the ordering according to the number of beds needed for each cluster. We tested these three constraints for both the exact formulation and the approximation solution method. The results of these tests are shown in Table 7.1 which shows the runtime in seconds, the obtained objective function value, and, in case the optimal solution is not found within ten hours, the LP-bound.

Exact method	Runtime (s)	Objective	LP-bound
# Clinical departments	36000	–	20
# Patients treated per day	1899	57	57
# Beds	6450	57	57
Approximation method	**Runtime (s)**	**Objective**	**LP-bound**
# Clinical departments	3290	57	57
# Patients treated per day	5609	57	57
# Beds	866	57	57

Table 7.1: Computational results symmetry-breaking constraints

The symmetry-breaking constraints do not have the same effect on both solution methods, because of the differences in the ILP formulations. The symmetry-breaking constraint that reduces the computation time the most for the exact formulation is the constraint that sorts the clusters according to the expected number of patients treated per day, i.e., constraint (7.20). The symmetry-breaking constraint that reduces the computation time the most for the approximation solution method is the constraint that sorts the clusters according to the number of beds needed, i.e., constraint (7.21). Therefore, we add constraints (7.20) and (7.21) to the ILP formulations of the exact and approximation solution method, respectively.

7.4.2 Parameter settings hybrid heuristic

For the hybrid heuristic, we have to determine several parameter values. The first parameter is N, the number of randomly generated clusters. As for the considered instance the number of feasible clusters is quite small, namely 575, we choose to consider all feasible clusters. As mentioned before, the minimum number of clusters needed for the instance is three. The first group consist of only the cardiology department, and therefore, there is only one feasible cluster for this group. The second group is the group of non-surgical departments. This group consists of six different departments, and thus, we can generate 2^6 different clusters for this group. The third group consist of the nine surgical departments which results in 2^9 extra clusters. In total this results in $1 + 2^6 + 2^9 = 575$ different feasible clusters.

Then, there are two remaining parameter values that need to be determined. The first is the number of clusters Q to select in each step and the second is the stopping criterion K. To determine the best values for these parameters, we performed several tests by solving the problem 25 times for several parameter settings. For each parameter setting, we have determined the number of times the optimal solution is found, the standard deviation of the objective values, and the minimum, average, and maximum runtime in seconds. The computational results are given in Table 7.2.

As expected, the runtime increase when Q or K increases as more iterations

Q	K	# Optimal	Std. dev. objective	Minimum runtime (s)	Average runtime (s)	Maximum runtime (s)
40	25	13	6.4	149	291	409
40	50	15	8.6	354	519	772
40	75	21	8.7	430	764	1409
40	100	19	2.6	572	828	1223
50	25	17	6.5	241	372	586
50	50	20	2.4	434	706	1303
50	75	23	1.6	690	1080	1870
50	100	23	1.6	775	1102	2040
60	25	19	6.5	321	507	813
60	50	22	1.9	631	1015	2674
60	75	25	0	948	1337	2047

Table 7.2: Parameter settings hybrid heuristic

are performed. In addition, the number of times the optimal solution is found also increases with Q and K. Therefore, we should make a trade-off between the runtime and the quality of the solution found. We choose to set Q to 60 and K to 75 as these settings provides the best quality of the solution found. In addition, the runtime for these settings is still reasonable. For the further computational tests, these values are chosen.

7.4.3 Comparing solution methods

To compare the three proposed solution methods, we have used the methods to solve the instance described in the introduction of this section with blocking probabilities 0.05 and 0.1. In this way, we also get a feeling about the influence of the blocking probability on the obtained results.

Tables 7.3 and 7.4 show the results for the three proposed solution methods for blocking probabilities 0.05 and 0.1, respectively. For each method and blocking probability combination, we show (1) the total computation time in seconds, (2) the objective function value, (3) the number of clusters used, (4) the maximum number of clinical departments clustered into one cluster, (5) the total distance between assigned wards, (6) the total fulfillment of preferences, (7) the number of beds needed according to the Erlang loss model, and (8) the number of beds needed according to the approximation.

When the blocking probability increases, less beds are needed by the clusters. Therefore, we expect that increasing the blocking probability will improve the achieved solution with respect to the objective function value. The results depicted in Tables 7.3 and 7.4 confirm this as the objective function value decreases from 57 to 36. This means that instead of clustering five clinical departments only

$B = 0.05$	Exact method	Approximation method	Hybrid heuristic
Time (s)	1899	866	1175
Objective	57	57	57
# Clusters	5	5	5
$\max_c \sum_d X_{dc}$	5	5	5
$\sum_{c,v,w} g_{wv} Y_{cw} Y_{cv}$	22	22	22
$\sum_{d,c,w} h_{dw} X_{dc} Y_{cw}$	15	15	15
# Beds exact	335	336	335
# Beds approx.	–	338	–

Table 7.3: Results all methods with blocking probability 0.05

three departments are clustered. This leads to an increase in the number of clusters formed. In addition, the total distance between assigned wards decreases from 22 to 16. The solution obtained when the blocking probability is set to 0.1 performs slightly worse when we look at fulfilling the preferences of a clinical department for a certain ward as this value decreases from 15 to 10.

$B = 0.1$	Exact method	Approximation method	Hybrid heuristic
Time (s)	$>$ 10 hours	1773	796
Objective	36	36	36
# Clusters	6	6	6
$\max_c \sum_d X_{dc}$	3	3	3
$\sum_{c,v,w} g_{wv} Y_{cw} Y_{cv}$	16	16	16
$\sum_{d,c,w} h_{dw} X_{dc} Y_{cw}$	10	10	10
# Beds exact	309	309	309
# Beds approx.	–	311	–

Table 7.4: Results all methods with blocking probability 0.1

Tables 7.3 and 7.4 show that all considered solution methods lead to the same objective function value. Thus, we can conclude that both heuristic solution methods perform well, and therefore, provide good solutions to the original problem.

However, there are also some differences between the solutions found by the three solution methods. All solutions have the same objective function value, but the formed clusters and the assignment of clusters to wards differ for each of the three methods. Thus, there are several configurations which lead to the same objective function value. Extra terms might be added to the objective function to make

Abbreviation	Description
CAR	Cardiology
DER	Dermatology
GAS	Gastroenterology
GEN	General surgery
GER	Geriatrics
GYN	Gynecology
INT	Internal medicine
NEU	Neurology

Abbreviation	Description
OPH	Ophthalmology
ORA	Orthognathic surgery
ORT	Orthopedics
OTO	Otolaryngology
PLA	Plastic surgery
PUL	Pulmonary medicine
RHE	Rheumatology
URO	Urology

Table 7.5: Abbreviations clinical departments

a distinction between these solutions. Figure 7.6 shows the obtained solutions for all three solution methods. The used abbreviations in Figure 7.6 are described in Table 7.5.

We also see differences between the three solution methods with respect to the computation time. The computation time for the exact solution method is the longest for both chosen values of the blocking probability when compared to the two heuristic solution methods. We also see that the computation time for the exact solution method increases significantly when the blocking probability is changed from 0.05 to 0.1. This indicates that even the order of magnitude of the computation time for the exact solution method cannot easily be predicted beforehand which makes this method somehow impractical. The computation times for the two heuristic methods also change with the blocking probability. When the blocking probability is set to 0.05 the approximation solution method is faster, but when the blocking probability is set to 0.1 the hybrid heuristic is faster. Because it is not clear from these results which of the two heuristic solution methods performs better overall, we have used both methods to evaluate the scenarios described in the next section.

7.4.4 Evaluating scenarios

As the average LOS μ and the expected number of admissions per day λ are very uncertain parameters which may change over the years, it is necessary to evaluate several what-if scenarios. These evaluations can be used to find a solution which is robust to foreseen changes in μ and λ.

For the considered instance, we only know $\mu\lambda$, i.e., the expected number of patients admitted per day, and not the values of μ and λ separately. Therefore, we cannot investigate changes in μ or λ, however, we can investigate a change in $\mu\lambda$.

Let us assume that the growth of each clinical department lies between 0.75 and 1.25, i.e., $\mu\lambda$ can reduce to at most $0.75\mu\lambda$ and increase to at most $1.25\mu\lambda$. Then, we

Exact method

	A	B
11	NEU-ORT OTO-PLA	NEU-ORT OTO-PLA
10	GER-INT RHE	GER-INT RHE
9		GER-INT RHE
8		
7	DER-GAS PUL	DER-GAS PUL
6	GEN-GYN OPH-ORA URO	GEN-GYN OPH-ORA URO
5	CAR	CAR
4	GEN-GYN OPH-ORA URO	CAR

Approximation method

	A	B
11	NEU-OPH ORA-PLA	NEU-OPH ORA-PLA
10	GEN-GYN ORT-OTO URO	GEN-GYN ORT-OTO URO
9		GEN-GYN ORT-OTO URO
8		
7	DER-GAS GER-PUL	DER-GAS GER-PUL
6	INT-RHE	INT-RHE
5	CAR	CAR
4	CAR	INT-RHE

Hybrid heuristic

	A	B
11	NEU-ORA OTO-PLA URO	NEU-ORA OTO-PLA URO
10	GEN-GYN OPH-ORT	GEN-GYN OPH-ORT
9		GEN-GYN OPH-ORT
8		
7	GAS-GER PUL	GAS-GER PUL
6	DER-INT RHE	DER-INT RHE
5	CAR	CAR
4	CAR	DER-INT RHE

Figure 7.6: Lay-out hospital 0.05 blocking probability

can determine the effect on the obtained solution when the blocking probability is set to 0.05. When $\mu\lambda$ reduces to $0.75\mu\lambda$ for some of the clinical departments and none of the other clinical departments grows, the obtained solution in Section 7.4.3 with blocking probability 0.05 is still feasible. Moreover, in many cases a better solution can be found because less beds are needed. Table 7.6 shows the results for the two heuristic solution methods when for each clinical department $\mu\lambda$ is reduced to $0.75\mu\lambda$.

Table 7.6 indeed shows that the objective function value is reduced from 57 to 20. This reduction is caused by the fact that the number of clinical departments clustered is reduced from five to three which in turn leads to an increase in the number of clusters formed. The solution obtained by the approximation solution method uses six clusters and the hybrid heuristic uses eight clusters. We also see that the distance between assigned wards is reduced to zero for all formed

$B = 0.05$	Approximation method	Hybrid heuristic
Time (s)	851	263
Objective	20	20
# Clusters	6	8
$\max_c \sum_d X_{dc}$	3	3
$\sum_{c,v,w} g_{wv} Y_{cw} Y_{cv}$	0	0
$\sum_{d,c,w} h_{dw} X_{dc} Y_{cw}$	10	10
# Beds exact	263	268
# Beds approx.	265	–

Table 7.6: Results when $\mu\lambda$ is reduced to $0.75\mu\lambda$

clusters, but that the preferences are fulfilled less as this value is reduced from 15 to 10. For this scenario, we see that the hybrid heuristic takes less time than the approximation solution method.

When some of the clinical departments grow, i.e., the value of $\mu\lambda$ increases, the solution obtained in Section 7.4.3 with $B = 0.05$ may become infeasible. This is the case when the number of beds needed by a cluster exceeds the number of beds available on the assigned wards. To see when this problem may occur, we present in Table 7.7 for all three solution methods the clinical departments assigned to a cluster, the number of beds needed by this cluster, and the number of beds assigned to this cluster.

Table 7.7 shows how much slack is available for each cluster. As slack we define the number of spare beds available at the assigned wards. Some clusters already need all assigned beds, but some have more than 20 extra beds available. The more spare beds are available, the more a cluster can grow in the coming years. However, when a cluster with no or only a few spare beds grows, the obtained solution might become infeasible. This does not necessarily mean that no feasible solution can be found, but that the generated clusters and the assignment of these clusters to wards may have to be changed. However, when too many clinical departments grow it might be the case that no feasible solution is available. For the considered case, there is no feasible solution when $\mu\lambda$ increases to $1.25\mu\lambda$ for all clinical departments. Therefore, in the following, we determined the maximum possible growth for all departments such that there is still a feasible solution available.

Note that the minimum number of clusters needed is three as described in the introduction of this section. The total number of beds needed is minimal when the clinical departments are clustered as much as possible. Thus, to determine the maximal growth of the clinical departments we consider this extreme solution with only three clusters. The first cluster only consists of the cardiology department, the second cluster consist of all remaining surgical departments, i.e., GEN, GYN, NEU, OPH, ORA, ORT, OTO, PLA, and URO, and the third cluster consists of all remain-

	Exact method	Approximation method	Hybrid heuristic
Cluster 1: Departments	INT-RHE-GER	INT-RHE	INT-RHE-DER
# Beds needed/available	80/96	80/88	80/88
Cluster 2: Departments	GEN-GYN-OPH ORA-URO	GEN-GYN-ORT OTO-URO	GEN-GYN-OPH ORT
# Beds needed/available	72/72	78/96	75/96
Cluster 3: Departments	CAR	CAR	CAR
# Beds needed/available	65/86	65/70	65/70
Cluster 4: Departments	GAS-PUL-DER	GAS-PUL-DER GER	GAS-PUL-GER
# Beds needed/available	63/64	63/64	63/64
Cluster 5: Departments	NEU-PLA-ORT OTO	NEU-PLA-OPH ORA	NEU-PLA-ORA OTO-URO
# Beds needed/available	55/60	50/60	52/60

Table 7.7: Solutions provided by solution methods with B set to 0.05 including number of beds needed/available

ing non-surgical departments, i.e., DER, GAS, GER, INT, PUL, and RHE. We have considered three scenarios. For each of the three scenarios, we allow maximal growth, i.e., $1.25\mu\lambda$, for two of the clusters and we determine the maximum allowed growth for the third cluster such that there is still a feasible solution. For the two clusters with maximal growth, we can determine the assignment of wards to these clusters such that a feasible solution is obtained with as few beds as possible. This means that the total slack on the assigned wards is minimized. The total number of beds available on the remaining wards is then available for the third cluster, and thus, we can determine the maximum allowed growth for this third cluster. When the maximum growth is determined, we use the two heuristic solution methods to find the optimal solution with the given input parameters $\mu\lambda$. The assignment of clusters to wards may be different than in the assignment determined beforehand, because a better solution may exist when the original objective function is considered.

Table 7.8 shows the results for these three scenarios. All of these three scenarios are infeasible according to the approximation method, because the number of beds needed for each cluster is overestimated. For the approximation method, the row '# Beds exact' denotes the number of beds needed when three clusters are used. The row '# Beds approx.' denotes the number of beds needed for the three clusters when the approximation function is used. Because the clusters are relatively

$B = 0.05$	Approximation method Scen. 1	Scen. 2	Scen. 3	Hybrid heuristic Scen. 1	Scen. 2	Scen. 3
Time (s)	–	–	–	228	209	333
Objective	Infeasible	Infeasible	Infeasible	165	93	112
# Clusters	–	–	–	3	4	4
$\max_c \sum_d X_{dc}$	–	–	–	9	6	6
$\sum_{c,v,w} g_{wv} Y_{cw} Y_{cv}$	–	–	–	90	48	62
$\sum_{d,c,w} h_{dw} X_{dc} Y_{cw}$	–	–	–	15	15	10
# Beds exact	369	363	365	369	370	373
# Beds approx.	378	371	375	–	–	–

Table 7.8: Results for the three maximum growth scenarios

large, the number of beds needed is overestimated more than was the case for the original instance. This phenomenon can easily be seen in Figure 7.4. Because of this overestimation for each of the three clusters, the total number of beds needed by the clusters lies quite close to the total number of beds available, i.e., 378, and therefore, no feasible solution can be found. The hybrid heuristic uses the exact method to determine the number of beds needed. Therefore, this number is not overestimated, and thus, a feasible solution can be found. For two of the scenarios, even a feasible solution with four clusters is found. Because the number of beds needed almost equals the number of beds available, not much flexibility is left to improve the assignment of clusters to wards. Therefore, the total distance between wards assigned to a cluster is quite large. These results show that when the clusters are relatively large the hybrid heuristic is preferred as this method is better in finding a feasible solution. In addition, the computation time is very short which makes this method applicable in practice, as several scenarios can be evaluated within a short period of time.

7.5 Conclusions and recommendations

We have introduced a problem, which in this combination has not been considered in literature before. We cluster one or more clinical departments to reduce the number of beds needed and assign these clusters to wards such that the resulting blocking probability is below a given threshold. Each clinical department is assigned to exactly one cluster such that the clusters form a partition of the clinical departments. Only one cluster is allowed to be assigned to a ward such that only clustered clinical departments share beds. We have developed an exact solution method and two heuristic solution methods to solve this problem.

The exact solution method uses the Erlang loss model to determine how many

beds are needed to guarantee a prespecified blocking probability. An ILP is used to determine which departments should be clustered and to assign wards to the formed clusters. A drawback of this exact solution method is that the computation time is unpredictable and quite long. Therefore, we have developed two heuristic solution methods that need less computation time, but still provide good solutions to the original problem.

One of the heuristic solution methods approximates the number of beds needed, and thereby, eliminates the evaluation of the complex Erlang loss formulation. Our tests show that this Erlang loss formulation can easily be approximated by a linear function that minimizes the maximum overestimation of the number of beds. Also for this approach an ILP is used to cluster several clinical departments and to assign the formed clusters to wards. As the number of beds needed by a cluster might be overestimated, we cannot guarantee that this method finds the optimal solution.

The other heuristic solution method uses the exact Erlang loss model to determine the number of beds needed, but uses a local search approach to determine which clusters should be formed. The assignment of clusters to wards is again done with the use of an ILP. In each iteration of the heuristic, a set of clusters is selected such that each clinical department is assigned to exactly one of these clusters. For this set of clusters, the optimal assignment of these clusters to the wards is determined. The number of beds needed for each cluster is determined beforehand with the use of the Erlang loss model. As the selection of the clusters is determined by a heuristic, this method does not guarantee to find the optimal solution.

All three considered solution methods use the Erlang loss formula or an approximation of this formula to determine the number of beds needed for each cluster. The input for these formulas are the average LOS and the expected number of admissions per day. However, these values are quite uncertain and may change over the years, and thereby, the number of beds needed by a cluster might increase or decrease. Therefore, it is important to evaluate several scenarios before fixing a new lay-out for the hospital.

Our computational results show that the exact solution method takes quite some time, and therefore, this method is not useful in practice. The computation time for the two other heuristic solution methods is much shorter and these methods are therefore better applicable in practice. However, the approximation solution method overestimates the number of beds needed for some cases, and thus, might claim that no feasible solution exists. But when the number of beds needed is determined exactly, it might be the case that there does exist a feasible solution. Concluding, we believe that the hybrid heuristic is the best solution method for solving this problem as several scenarios can be evaluated within a short amount of time. In addition, the hybrid heuristic always finds a feasible solution when a feasible solution exists. The only drawback is that we cannot guarantee that the optimal solution is found.

A disadvantage of using the Erlang loss formula may be that it only uses the

expected number of admission per day. One may claim, that it therefore can only be used in theory, because the number of admissions per day varies during the week. This would mean that the number of beds needed in practice is higher than is determined by the Erlang loss formula. However, several methods exist that level the bed occupancy over the week such that peak occupancies can be reduced or even avoided, see e.g. Chapter 6. Thus, the number of beds determined by the Erlang loss model should be enough in practice.

As the input parameters μ and λ are quite uncertain, further research should focus on finding robust solutions. As a robust solution we define a solution which is still feasible when the input parameters change in a reasonable way. A possible solution would be to look at a worse case scenario, however, this will lead to a suboptimal solution when the input parameters have different values than in this worst case scenario. Thus, further research should focus on finding solutions which can easily be adapted such that they are near optimal for many or all reasonable values of the input parameters.

Epilog

In this dissertation, we discuss various topics concerning planning in healthcare. In each of the chapters, some kind of uncertainties are taken into account as uncertainty is one of the main characteristics of planning in healthcare. However, including these uncertainties increases the complexity of the developed models, and therefore, a trade-off has to be made between creating more realistic or less complex models.

In Chapter 3, we have discussed ambulance planning on the strategic and tactical level. Both levels can be planned simultaneously by solving a proposed stochastic program. In addition, we provided solution approaches for solving both levels separately, namely an Integer Linear Program (ILP) for the strategic level, and simulation combined with a local search procedure for the tactical level. One main advantage of the presented formulations is that the coverage constraints are modeled in a generic way, and thus, can easily be adapted to different requirements. Therefore, the formulations are not only applicable for the Netherlands but for many countries. The computational results show that the solution approach that solves both levels simultaneously is more suitable to make a good trade-off between minimizing the number of bases and ambulances. However, the computation time of this approach increases exponentially whereas the computation time of the approach that solves both levels separately follows a more linear trend. As a next step, it might be interesting to test our models on different Emergency Medical Service (EMS) systems to see if the developed models are as generic as we claim them to be or if we need to make further generalizations. In addition, we aim to create models for the operational level to be able to solve the ambulance planning on all levels. Solving the operational level also helps to determine the coverage in practice. In addition, extra modifications on the operational level such as relocating ambulances might further reduce the number of ambulances needed.

Another problem addressed in this dissertation is minimizing the waiting time of emergency surgeries which is discussed in Chapter 4. In this chapter, we have introduced the 'break-in-moments' (BIM) problem that deals with sequencing elective surgeries, which are preassigned to an Operating Room (OR), to reduce emergency surgery waiting time. This is a new type of scheduling problem which has not yet been discussed in literature. We developed and tested several heuristic solution methods to solve the problem. The results of the computational study show that the expected maximum break-in-interval (BII) can be reduced by more than 20% for the schedule used at the Erasmus Medical Center. The simulation results show that it is beneficial to take the maximum BII into account when creating a schedule, because our heuristics outperform the original schedule when

compared to the expected and simulated average maximum BII. However, when an emergency surgery breaks in into the schedule, the OR-schedule is disturbed. To increase the stability of the OR-schedule, it can be beneficial to insert breaks between surgeries. The simplest way to deal with disturbances caused by uncertain surgery durations is to insert a break in the middle of the day. In this way, surgeries scheduled after the midday break can mostly start on time, even if surgeries earlier that day have taken longer than expected. Furthermore, during a break, emergency surgeries can start immediately. Therefore, it is preferable to spread the breaks of the different ORs a bit over the day, such that the overlap in breaks of ORs is small. Another approach can be to plan small breaks at the end of a surgery whose duration has a high variance. Incorporating this into the BIM problem and further simulation experiments should clarify the effect of BIM optimization when these considerations are taken into account.

When the OR-schedule is disturbed during the day it might be necessary to reschedule the surgeries. In Chapter 5, we formulated an ILP for this problem which determines the best adjusted OR-schedule at a given point in time. The results show that patients, wards, and OR assistants have opposite interests compared to the recovery, radiology, pathology, and logistic department. Furthermore, the achieved results show that, with a few exceptions, the only used adjustments are (i) shifting surgeries, and (ii) scheduling breaks between two surgeries. These two decision rules are incorporated in a developed decision support system (DSS). This system determines the best adjusted schedule for one OR with respect to the given constraints and gives insight in how the workload of stakeholders is influenced by adjusting the OR-schedule throughout the day. The simulation study shows that by using this DSS, less surgeries are canceled and patients and wards are more satisfied, but also that the workload of several departments increases to compensate this. The developed DSS can be used in several ways, for example, reschedule an OR immediately when it is disturbed or reschedule all ORs at some moments in time. The last option also raises the question in what order the ORs should be rescheduled. Furthermore, it would be interesting to investigate the best way to use the DSS in practice.

Often the number of beds at hospitals is limited. Therefore, we introduce in Part IV two ways to deal with this. In Chapter 6, we introduce two approaches that reduce the number of required beds by adjusting the OR-schedule. The first approach incorporates an analytical formulation of the probability distribution of the bed occupancy and improves the OR-schedule by using a local search procedure. The second approach approximates the required number of beds by the expected bed occupancy, which enables us to solve the problem as an ILP. The computational results show that the ILP with the simplified objective function performs the best for instances based on the situation in HagaZiekenhuis. Note that after solving the ILP, the number of required beds is still determined by using the analytic formulation. The computational results show that the number of required beds at the orthopedic department of HagaZiekenhuis can be reduced by almost 20% based on the results of the ILP. The approach developed in this chapter can

also be used to include emergency surgeries by introducing dummy OR-blocks that are already fixed to a specific day in the planning horizon and contain the expected number of emergency surgeries. In this way, the arrival and admission of emergency patients is considered while determining a new OR-schedule for the elective surgeries. Thus, the total number of required beds is minimized and both elective and emergency patients can be admitted after surgery. To further reduce the number of beds needed, we should also include creating the OR-blocks into our modeling approach. This would impose some extra constraints on the model, because we also have to consider the stochastic duration of the surgeries such that the required surgical time does not exceed the available surgical time. Thus, it would be interesting to investigate this problem in future research.

In Chapter 7, we deal with a limited bed capacity by clustering one or more clinical departments. After creating the clusters, the clusters have to be assigned to wards such that the resulting blocking probability is below a given threshold. Each clinical department is assigned to exactly one cluster such that the clusters form a partition of the clinical departments. Only one cluster is allowed to be assigned to a ward such that only clustered clinical departments share beds. We have developed an exact solution method and two heuristic solution methods to solve this problem. The exact solution method uses the Erlang loss model to determine how many beds are needed to guarantee a prespecified blocking probability. An ILP is used to determine which departments should be clustered and to assign wards to the formed clusters. A drawback of this exact solution method is that the computation time is unpredictable and quite long. Therefore, we have developed two heuristic solution methods that need less computation time, but still provide good solutions to the original problem. One of the heuristic solution methods approximates the number of beds needed by a linear function, thereby eliminating the evaluation of the complex Erlang loss formulation. Also for this approach an ILP is used to cluster several clinical departments and to assign the formed clusters to wards. As the number of beds needed by a cluster might be overestimated, we cannot guarantee that this method finds an optimal solution. The other heuristic solution method uses the exact Erlang loss model to determine the number of beds needed, but uses a local search approach to determine which clusters should be formed. The assignment of clusters to wards is again done with the use of an ILP. As the selection of the clusters is determined by a heuristic, this method also does not guarantee to find an optimal solution. Further research should focus on finding robust solutions as the average length of stay and the expected number of admissions per day, which are used as input parameters, are quite uncertain. Hereby, a robust solution is a solution which is still feasible when the input parameters change in a reasonable way. A possible solution would be to look at a worse case scenario, however, this will lead to a suboptimal solution when the input parameters have different values than in this worst case scenario. Thus, further research should focus on finding solutions which can easily be adapted such that they are near optimal for many or all reasonable values of the input parameters.

Concluding, this dissertation gives several examples, where including uncer-

tainties in the modeling process leads to complex models that need to be simplified in order to solve them in a reasonable amount of time. Moreover, the considered problems are often already $\mathcal{NP}$-hard without including uncertainty. To still be able to solve the problems, we either approximate some part of the model or use heuristics to solve the problem. Both approaches lead to suboptimal solutions as the first approach does not solve the original problem and the second approach often ends in a local optimum instead of a global optimum. Nevertheless, the developed approaches form a good first step in solving these complex problems.

Bibliography

[1] E. Aarts and J. Korst. Simulated annealing and Boltzmann machines: a stochastic approach to combinatorial optimization and neural computing. *John Wiley & Sons*, 1989.

[2] J. Adams, E. Balas, and D. Zawack. The shifting bottleneck procedure for job shop scheduling. *Management Science*, 34 (3): 391 – 401, 1988.

[3] I. Adan, J. Bekkers, N. Dellaert, J. Jeunet, and J. Vissers. Improving operational effectiveness of tactical master plans for emergency and elective patients under stochastic demand and capacitated resources. *European Journal of Operational Research*, 213 (1): 290 – 308, 2011.

[4] I. Adan, J. Bekkers, N. Dellaert, J. Vissers, and X. Yu. Patient mix optimisation and stochastic resource requirements: a case study in cardiothoracic surgery planning. *Health Care Management Science*, 12 (2): 129 – 141, 2009.

[5] A.A. Aly and J.A. White. Probabilistic formulation of the emergency service location problem. *The Journal of the Operational Research Society*, 29 (12): 1167 – 1179, 1978.

[6] R. Bekker and P.M. Koeleman. Scheduling admissions and reducing variability in bed demand. *Health Care Management Science*, 14 (3): 237 – 249, 2011.

[7] J. Beliën and E. Demeulemeester. Building cyclic master surgery schedules with leveled resulting bed occupancy. *European Journal of Operational Research*, 176 (2): 1185 – 1204, 2007.

[8] J. Beliën, E. Demeulemeester, and B. Cardoen. A decision support system for cyclic master surgery scheduling with multiple objectives. *Journal of Scheduling*, 12 (2): 147 – 161, 2009.

[9] P. Beraldi and M.E. Bruni. A probabilistic model applied to emergency service vehicle location. *European Journal of Operational Research*, 196 (1): 323 – 331, 2009.

[10] P. Beraldi, M.E. Bruni, and D. Conforti. Designing robust emergency medical service via stochastic programming. *European Journal of Operational Research*, 158 (1): 183 – 193, 2004.

[11] G.N. Berlin and J.C. Liebman. Mathematical analysis of emergency ambulance location. *Socio-Economic Planning Sciences*, 8 (6): 323 – 328, 1974.

[12] O. Berman, Z. Drezner, and D. Krass. Discrete cooperative covering problems. *Journal of the Operational Research Society*, 62 (11): 2002 – 2012, 2011.

[13] J. Bowers and G. Mould. Managing uncertainty in orthopaedic trauma theatres. *European Journal of Operational Research*, 154 (3): 599 – 608, 2004.

[14] L. Brotcorne, G. Laporte, and F. Semet. Ambulance location and relocation models. *European Journal of Operational Research*, 147 (3): 451 – 463, 2003.

[15] F. Busetti. Simulated annealing overview. *http://aiinfinance.com/saweb.pdf*, 2003.

[16] B. Cardoen, E. Demeulemeester, and J. Beliën. Optimizing a multiple objective surgical case sequencing problem. *International Journal of Production Economics*, 119 (2): 354 – 366, 2009.

[17] B. Cardoen, E. Demeulemeester, and J. Beliën. Sequencing surgical cases in a day-care environment: an exact branch-and-price approach. *Computers & Operations Research*, 36 (9): 2660 – 2669, 2009.

[18] B. Cardoen, E. Demeulemeester, and J. Beliën. Operating room planning and scheduling: a literature review. *European Journal of Operational Research*, 201 (3): 921 – 932, 2010.

[19] S.C. Chapman and J.A. White. Probabilistic formulations of emergency service facilities location problems. In *ORSA/TIMS Conference, San Juan, Puerto Rico*, 1974.

[20] V.S. Chow, M.L. Puterman, N. Salehirad, W. Huang, and D. Atkins. Reducing surgical ward congestion through improved surgical scheduling and uncapacitated simulation. *Production and Operations Management*, 20 (3): 418 – 430, 2011.

[21] R. Church and C. ReVelle. The maximal covering location problem. *Papers in Regional Science*, 32 (1): 101 – 118, 1974.

[22] S.A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing*, 151 – 158, 1971.

[23] G.B. Dantzig and M.N. Thapa. Linear programming 1: introduction. *Springer*, 1997.

[24] G.B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8 (1): 101 – 111, 1960.

[25] M.S. Daskin. A maximum expected covering location model: formulation, properties and heuristic solution. *Transportation Science*, 17 (1): 48 – 70, 1983.

[26] A. de Bruin, R. Bekker, L. van Zanten, and G. Koole. Dimensioning hospital wards using the Erlang loss model. *Annals of Operations Research*, 178 (1): 23 – 43, 2010.

[27] B. Denton, J. Viapiano, and A. Vogl. Optimization of surgery sequencing and scheduling decisions under uncertainty. *Health Care Management Science*, 10 (1): 13 – 24, 2007.

[28] F. Dexter. A strategy to decide whether to move the last case of the day in an operating room to another empty operating room to decrease overtime labor costs. *Anesthesia & Analgesia*, 91 (4): 925 – 928, 2000.

[29] F. Dexter, J.D. Lee, A.J. Dow, and D.A. Lubarsky. A psychological basis for anesthesiologists' operating room managerial decision-making on the day of surgery. *Anesthesia & Analgesia*, 105 (2): 430 – 434, 2007.

[30] F. Dexter, A. Macario, and R. Traub. Optimal sequencing of urgent surgical cases. *Journal of Clinical Monitoring and Computing*, 15 (3): 153 – 162, 1999.

[31] F. Dexter, A. Willemsen-Dunlap, and J. Lee. Operating room managerial decision-making on the day of surgery with and without computer recommendations and status displays. *Anesthesia & Analgesia*, 105 (2): 419 – 429, 2007.

[32] J. Dréo, A. Pétrowski, P. Siarry, and E. Taillard. Metaheuristics for hard optimization. *Springer*, 2006.

[33] H. Fei, N. Meskens, and C. Chu. A planning and scheduling problem for an operating theatre using an open scheduling strategy. *Computers & Industrial Engineering*, 58 (2): 221 – 230, 2010.

[34] O. Fujiwara, T. Makjamroen, and K.K. Gupta. Ambulance deployment analysis: a case study of Bangkok. *European Journal of Operational Research*, 31 (1): 9 – 18, 1987.

[35] R.D. Galvão, F.Y. Chiyoshi, and R. Morabito. Towards unified formulations and extensions of two classical probabilistic location models. *Computers and Operations Research*, 32 (1): 15 – 33, 2005.

[36] M.R. Garey and D.S. Johnson. Computers and intractability: a guide to the theory of $\mathcal{NP}$-completeness. *W. H. Freeman & Co Ltd*, 1979.

[37] M. Gendreau, G. Laporte, and F. Semet. Solving an ambulance location model by tabu search. *Location Science*, 5 (2): 75 – 88, 1997.

[38] Y. Gerchak, D. Gupta, and M. Henig. Reservation planning for elective surgery under uncertain demand for emergency surgery. *Management Science*, 42 (3): 321 – 334, 1996.

[39] N. Geroliminis, M.G. Karlaftis, and A. Skabardonis. A spatial queuing model for the emergency vehicle districting and location problem. *Transportation Research Part B: Methodological*, 43 (7): 798 – 811, 2009.

[40] F. Glover and M. Laguna. Tabu search. *Kluwer Academic Publishers*, 1997.

[41] J. Goldberg, R. Dietrich, J.M. Chen, M. Mitwasi, T. Valenzuela, and E. Criss. A simulation model for evaluating a set of emergency vehicle base locations: development, validation, and usage. *Socio-Economic Planning Sciences*, 24 (2): 125 – 141, 1990.

[42] F. Gorunescu, S.I. McClean, and P.H. Millard. A queueing model for bed-occupancy management and planning of hospitals. *Journal of the Operational Research Society*, 53 (1): 19 – 24, 2002.

[43] F. Gorunescu, S.I. McClean, and P.H. Millard. Using a queueing model to help plan bed allocation in a department of geriatric medicine. *Health Care Management Science*, 5 (4): 307 – 312, 2002.

[44] L.V. Green and V. Nguyen. Strategies for cutting hospital beds: the impact on patient service. *Health Services Research*, 36 (2): 421 – 442, 2001.

[45] E. Hans, G. Wullink, M. van Houdenhoven, and G. Kazemier. Robust surgery loading. *European Journal of Operational Research*, 185 (3): 1038 – 1050, 2008.

[46] E.W. Hans, M. Houdenhoven, and P.J.H. Hulshof. A framework for healthcare planning and control. In *Handbook of Healthcare System Scheduling*, *International Series in Operations Research & Management Science* 168, 303 – 320. Springer US, 2012.

[47] S.I. Harewood. Emergency ambulance deployment in Barbados: a multi-objective approach. *Journal of the Operational Research Society*, 53 (2): 185 – 192, 2002.

[48] P.R. Harper and A.K. Shahani. Modelling for the planning and management of bed capacities in hospitals. *Journal of the Operational Research Society*, 53 (1): 11 – 18, 2002.

[49] W. Hartholt. Beslissingsondersteuning voor het aanpassen van de online ok-planning. *Master's thesis, University of Twente*, 2010.

[50] S. Henderson and A. Mason. Ambulance service planning: simulation and data visualisation. In *Operations Research and Health Care*, *International Series in Operations Research and Management Science* 70, 77 – 102. Springer US, 2005.

[51] P.J.H. Hulshof, R.J. Boucherie, J.T. van Essen, E.W. Hans, J.L. Hurink, N. Kortbeek, N. Litvak, P.T. Vanberkel, E. van der Veen, B. Veltman, I.M.H. Vliegen, and M.E. Zonderland. ORchestra: an online reference database of OR/MS literature in health care. *Health Care Management Science*, 14 (4): 383 – 384, 2011.

[52] P.J.H. Hulshof, N. Kortbeek, R.J. Boucherie, E.W. Hans, and P.J.M. Bakker. Taxonomic classification of planning decisions in health care: a structured review of the state of the art in OR/MS. *Health Systems*, 1 (2): 129 – 175, 2012.

[53] A.P. Iannoni and R. Morabito. A multiple dispatch and partial backup hypercube queuing model to analyze emergency medical systems on highways. *Transportation Research Part E: Logistics and Transportation Review*, 43 (6): 755 – 771, 2007.

[54] A.P. Iannoni, R. Morabito, and C. Saydam. Optimizing large-scale emergency medical system operations on highways using the hypercube queuing model. *Socio-Economic Planning Sciences*, 45 (3): 105 – 117, 2011.

[55] O. Karasakal and E.K. Karasakal. A maximal covering location model in the presence of partial coverage. *Computers and Operations Research*, 31 (9): 1515 – 1526, 2004.

[56] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220 (4598): 671 – 680, 1983.

[57] A. Kokangul. A combination of deterministic and stochastic approaches to optimize bed capacity in a hospital unit. *Computer Methods and Programs in Biomedicine*, 90 (1): 56 – 65, 2008.

[58] G.J. Kommer, A.A. van der Veen, W.F. Botter, and I. Tan. Ambulances binnen bereik: analyse van de spreiding en beschikbaarheid van de ambulancezorg in nederland. *Technical Report, National Institute for Public Health and the Environment*, 2003.

[59] M. Lamiri, X. Xie, A. Dolgui, and F. Grimaud. A stochastic model for operating room planning with elective and emergency demand for surgery. *European Journal of Operational Research*, 185 (3): 1026 – 1037, 2008.

[60] S.D. Lapierre, D. Goldsman, R. Cochran, and J. DuBow. Bed allocation techniques based on census data. *Socio-Economic Planning Sciences*, 33 (1): 25 – 38, 1999.

[61] R.C. Larson. A hypercube queuing model for facility location and redistricting in urban emergency services. *Computers and Operations Research*, 1 (1): 67 – 95, 1974.

[62] P. Lebowitz. Schedule the short procedure first to improve OR efficiency. *AORN*, 78 (4): 651 – 659, 2003.

[63] J. Ledolter, F. Dexter, and R.E. Wachtel. Control chart monitoring of the numbers of cases waiting when anesthesiologists do not bring in members of call team. *Anesthesia & Analgesia*, 111 (1): 196 – 203, 2010.

[64] X. Li, P. Beullens, D. Jones, and M. Tamiz. An integrated queuing and multiobjective bed allocation model with application to a hospital in China. *Journal of the Operational Research Society*, 60 (3): 330 – 338, 2009.

[65] X. Li, Z. Zhao, X. Zhu, and T. Wyatt. Covering models and optimization techniques for emergency response facility location and planning: a review. *Mathematical Methods of Operations Research*, 74 (3): 281 – 310, 2011.

[66] E. Marcon and F. Dexter. Impact of surgical sequencing on post anesthesia care unit staffing. *Health Care Management Science*, 9 (1): 87 – 98, 2006.

[67] E. Marcon and F. Dexter. An observational study of surgeons' sequencing of cases and its impact on postanesthesia care unit and holding area staffing requirements at hospitals. *Anesthesia & Analgesia*, 105 (1): 119 – 126, 2007.

[68] V. Marianov and C. ReVelle. Siting emergency services. In *Facility location: a survey of applications and methods*, 203 – 227. Springer-Verlag, 1995.

[69] V. Marianov and C. ReVelle. The queueing maximal availability location problem: a model for the siting of emergency vehicles. *European Journal of Operational Research*, 93 (1): 110 – 120, 1996.

[70] R.A. Marjamaa, P.M. Torkki, E.J. Hirvensalo, and O.A. Kirvelä. What is the best workflow for an operating room? A simulation study of five scenarios. *Health Care Management Science*, 12 (2): 142 – 146, 2009.

[71] Z. Michalewicz and D.B. Fogel. How to solve it: modern heuristics. *Springer*, 2004.

[72] A. Nederland. Ambulances in-zicht 2011. *Technical Report*, *National Institute for Public Health and the Environment*, 2011.

[73] N. Noyan. Alternate risk measures for emergency medical service system design. *Annals of Operations Research*, 181 (1): 559 – 589, 2010.

[74] A. Oversberg. Minimizing the waiting time for emergency surgery. *Master's thesis*, *Universität Bonn*, 2010.

[75] S.H. Owen and M.S. Daskin. Strategic facility location: a review. *European Journal of Operational Research*, 111 (3): 423 – 447, 1998.

[76] C.H. Papadimitriou and K. Steiglitz. Combinatorial optimization: algorithms and complexity. *Prentice-Hall, Inc.*, 1982.

[77] D.N. Pham and A. Klinkert. Surgical case scheduling as a generalized job shop scheduling problem. *European Journal of Operational Research*, 185 (3): 1011 – 1025, 2008.

[78] M. Pinedo. Scheduling: theory, algorithms and systems. *Prentice Hall*, 2006.

[79] S. Qiao and L. Qiao. A robust and efficient algorithm for evaluating Erlang-B formula. *Technical Report*, 1998.

[80] C. ReVelle and K. Hogan. A reliability-constrained siting model with local estimates of busy fractions. *Environment and Planning B: Planning and Design*, 15 (2): 143 – 152, 1988.

[81] C. ReVelle and K. Hogan. The maximum availability location problem. *Transportation Science*, 23 (3): 192 – 200, 1989.

[82] V. Shakhov. Simple approximation for Erlang B formula. In *2010 IEEE Region 8 International Conference on Computational Technologies in Electrical and Electronics Engineering (SIBIRCON)*, 220 – 222, 2010.

[83] F. Silva and D. Serra. Locating emergency services with different priorities: the priority queuing covering location problem. *Journal of the Operational Research Society*, 59 (9): 1229 – 1238, 2008.

[84] F.C.R. Spieksma, G.J. Woeginger, and Z. Yu. Scheduling with safety distances. *Annals of Operations Research*, 57 (1): 251 – 264, 1995.

[85] P.S. Stepaniak, G.H.H. Mannaerts, M. de Quelerij, and G. de Vries. The effect of the operating room coordinator's risk appreciation on operating room efficiency. *Anesthesia & Analgesia*, 108 (4): 1249 – 1256, 2009.

[86] C. Swoveland, D. Uyeno, I. Vertinsky, and R. Vickson. A simulation-based methodology for optimization of ambulance service policies. *Socio-Economic Planning Sciences*, 7 (6): 697 – 703, 1973.

[87] R.A. Takeda, J.A. Widmer, and R. Morabito. Analysis of ambulance decentralization in an urban emergency medical service using the hypercube queueing model. *Computers and Operations Research*, 34 (3): 727 – 741, 2007.

[88] H.C. Tijms. A first course in stochastic models. *John Wiley & Sons*, 2003.

[89] C. Toregas, R. Swain, C. ReVelle, and L. Bergman. The location of emergency service facilities. *Operations Research*, 19 (6): 1363 – 1373, 1971.

[90] M. Utley, S. Gallivan, T. Treasure, and O. Valencia. Analytical methods for calculating the capacity required to operate an effective booked admissions policy for elective inpatient services. *Health Care Management Science*, 6 (2): 97 – 104, 2003.

[91] M. van Houdenhoven, J.M. van Oostrum, G. Wullink, E. Hans, J.L. Hurink, J. Bakker, and G. Kazemier. Fewer intensive care unit refusals and a higher capacity utilization by using a cyclic surgical case schedule. *Journal of Critical Care*, 23 (2): 222 – 226, 2008.

[92] P.J. van Laarhoven and E.H. Aarts. Simulated annealing: theory and applications. *Springer Publishing Company, Inc.*, 2010.

[93] J.M. van Oostrum, M. van Houdenhoven, J.L. Hurink, E.W. Hans, G. Wullink, and G. Kazemier. A master surgical scheduling approach for cyclic scheduling in operating room departments. *OR Spectrum*, 30 (2): 355 – 374, 2008.

[94] P.T. Vanberkel, R.J. Boucherie, E.W. Hans, J.L. Hurink, W.A.M. van Lent, and W.H. van Harten. Accounting for inpatient wards when developing master surgical schedules. *Anesthesia & Analgesia*, 112 (6): 1472 – 1479, 2011.

[95] P.T. Vanberkel, R.J. Boucherie, E.W. Hans, J.L. Hurink, W.A.M. van Lent, and W.H. van Harten. An exact approach for relating recovering surgical patient workload to the master surgical schedule. *Journal of the Operational Research Society*, 62 (10): 1851 – 1860, 2011.

[96] H.P. Williams. Model building in mathematical programming. *John Wiley & Sons*, 1999.

[97] G. Wullink, M. Van Houdenhoven, E.W. Hans, J. van Oostrum, M. van der Lans, and G. Kazemier. Closing emergency operating rooms improves efficiency. *Journal of Medical Systems*, 31 (6): 543 – 546, 2007.

[98] A.S. Zaki, H.K. Cheng, and B.R. Parker. A simulation model for the analysis and management of an emergency service system. *Socio-Economic Planning Sciences*, 31 (3): 173 – 189, 1997.

Acronyms

BII	Break-In-Interval
BIM	Break-In-Moment
CAR	Cardiology
CSSD	Central Sterile Supply Department
DER	Dermatology
DSM	Double Standard Model
DSS	Decision Support System
EMS	Emergency Medical Service
F-B	Forward-Backward
FLEX	Flexible goal values
GAS	Gastroenterology
GEN	General surgery
GER	Geriatrics
GYN	Gynecology
ICC	Integrated Change Constraints
ILP	Integer Linear Program
INT	Internal medicine
LB	Lower Bound
LOS	Length Of Stay
LP	Linear Program
LPT	Longest Processing Time first
LSCM	Location Set Covering Model
MALP	Maximum Availability Location Problem
MCLP	Maximal Cover Location Problem
MEXCLP	Maximum Expected Covering Location Problem
MIP	Mixed Integer Program
NEU	Neurology
OK	Operatiekamer
OPH	Ophthalmology
OR	Operating Room
ORA	Orthognathic surgery
ORT	Orthopedics
OS	Original Schedule
OTO	Otolaryngology
PLA	Plastic surgery
PUL	Pulmonary medicine
RHE	Rheumatology

SA Simulated Annealing
SAP Strategic Ambulance Planning
SBH Shifting Bottleneck Heuristic
SIPC Stochastic Integer problem formulation under Probabilistic Constraints
SPAB Stochastic Planning of Ambulances and Bases
SPT Shortest Processing Time first
TAP Tactical Ambulance Planning
TS Tabu Search
URO Urology

Summary

Due to an aging population and increased healthcare costs, hospitals are forced to use their resources more efficiently, meaning that the same number of patients has to be treated with less resources or more patients with the same amount of resources. The mentioned resources are, for example, the Operating Room (OR), the beds at the wards, and ambulances. Improving the efficiency in hospitals is challenging as many uncertainties have to be taken into account, e.g., the arrival of patients, the length of stay of patients, and the clinical path the patient follows. By carefully scheduling and planning the processes at the departments patients visit during their stay the efficiency can be improved. More specific, this means that procedures or appointments should be scheduled such that (1) the waiting time for the patients is minimized, (2) the schedule is robust against changes in the procedure or appointment time, and (3) the arrival of emergency patients is accounted for. In this dissertation, we focus on solution methods that take the three mentioned points into account. For some problems, one or two of these objectives are more important than the others and sometimes the objectives are even conflicting. In those cases, a balanced trade-off between the (conflicting) objectives has to be made. One possible way to deal with this conflict is to provide a weight for each objective such that the objectives are considered in a correct way.

This dissertation consists of four parts. The first part, consisting of Chapter 1 and 2, serves as an introduction. Chapter 1 discusses the challenges of improving the efficiency in healthcare and Chapter 2 introduces and explains the field of combinatorial optimization and the solution methods used in this dissertation.

In the remainder of the dissertation, problems are studied that originated at three hospitals in the Netherlands, namely HagaZiekenhuis, Isala Clinics, and Erasmus Medical Center. The problems considered all involve planning but occur on different planning levels. The problems are linked in the way that they all occur on a possible path through the hospital for an emergency patient that needs to undergo surgery.

Part II of this dissertation discusses problems concerning ambulance planning. Ambulance planning is done on different levels, namely the strategic, tactical, and operational level. In Chapter 3, we discuss solution methods for locating the ambulance bases on the strategic level and allocating ambulances to these bases on the tactical level. We present possible formulations for the problems and discuss solution approaches that solve both levels either simultaneously or separately. The models are set up such that various types of coverage constraints can be incorporated. Therefore, the models can be applied to different emergency medical services systems all over the world. The approaches are tested on data based on

the situation in the Netherlands. The results show that the solution approach that solves both levels separately performs better when considering minimizing the number of bases. However, the solution approach that solves both levels simultaneously performs better when considering minimizing the number of ambulances. In addition, with the latter solution approach it is easier to make a good trade-off between minimizing the number of bases and ambulances because it considers a weighted objective function. However, the computation time of this approach increases exponentially with the input size whereas the computation time of the approach that solves both levels separately follows a more linear trend.

Part III discusses problems concerning operating room planning. Chapter 4 focuses on performing emergency surgeries as quickly as possible. This is important, because postponing them generally increases a patient's risk of complications. We consider the case where emergency surgeries are scheduled in one of the elective ORs. In this situation, emergency patients are operated once an ongoing elective surgery has finished. We denote these completion times of the elective surgeries by 'break-in-moments' (BIMs). The waiting time for emergency surgeries can be reduced by spreading these BIMs as evenly as possible over the day. This can be achieved by sequencing the surgeries in their assigned OR, such that the maximum interval between two consecutive BIMs is minimized. We discuss several exact and heuristic solution methods for this new type of scheduling problem. In practice, the initial schedule may be disrupted by emergency surgeries arising throughout the day and the uncertainty of the durations of elective surgeries. As a result, the completion times of the elective surgeries, and therefore, the BIMs change, leading also to a change of the maximum distance between two BIMs. To estimate this effect and investigate the robustness of the created schedules, we conduct a simulation study. Computational results show that one of the developed algorithms reduces the waiting time of emergency surgeries by approximately 10% when compared to the schedule currently used at the Erasmus Medical Center.

Chapter 5 of Part III discusses the problem of rescheduling surgeries throughout the day. Due to surgery duration variability and arrivals of emergency surgeries, the planned OR-schedule is disrupted which may lead to changes in the start times of the elective surgeries. These changes may result in undesirable situations for patients, wards, or other involved departments, and therefore, the OR-schedule has to be adjusted. We develop a decision support system (DSS) which assists the OR manager in this decision by providing the three best adjusted OR-schedules. The system considers the preferences of all involved stakeholders and only evaluates the OR-schedules that satisfy the imposed resource constraints. The decision rules used for this system are based on a thorough analysis of the OR rescheduling problem. We model this problem as an Integer Linear Program (ILP) whose objective is to minimize the deviation from the preferences of the considered stakeholders. By applying this ILP to instances from Isala Clinics, we observed that the given preferences mainly lead to (i) shifting a surgery and (ii) scheduling a break between two surgeries. By using these changes in the DSS, the performed simulation study shows that less surgeries are canceled and patients and wards are more satisfied,

but also that the perceived workload of several departments increases to compensate this. The system can also be used to judge the acceptability of a proposed initial OR-schedule.

Part IV focuses on ward planning. In Chapter 6, methods are introduced for creating an OR-schedule that spreads the usage of beds nicely over time, and thereby, minimizes the number of required beds. An OR-schedule is given by an assignment of OR-blocks to specific days in the planning horizon and has to fulfill several resource constraints. Due to the stochastic nature of the length of stay of patients, the analytic calculation of the number of required beds for a given OR-schedule is a complex task involving the convolution of discrete distributions. We present two approaches to deal with this complexity. First, a heuristic approach based on local search is given that takes into account the detailed formulation of the objective. A second approach reduces the complexity by simplifying the objective function. This allows modeling and solving the resulting problem as an ILP. Both approaches are tested on data provided by HagaZiekenhuis in the Netherlands. Furthermore, several what-if scenarios are evaluated. The computational results show that the approach that uses the simplified objective function provides better solutions to the original problem for instances based on the situation in HagaZiekenhuis. By using this approach, the number of required beds for the considered instance of HagaZiekenhuis can be reduced by almost 20%.

Chapter 7 shows that when the number of available beds in a hospital is limited, it can be beneficial to cluster several clinical departments. However, not all clinical departments can be clustered as, e.g., surgical patients can easily get an infection from one of the non-surgical patients. In addition, patients from one clinical department should not be spread out over the entire hospital as this complicates the process of doing rounds. We consider a situation where wards with a fixed number of beds are given. The question is how to cluster the clinical departments and to determine the assignment of these clustered departments to the available wards such that enough beds are available to guarantee a blocking probability below a prespecified threshold. We first give an exact formulation of the problem to be able to achieve optimal solutions. However, computational experiments show that the resulting computation times for this method are too long to be applicable in practice. To reduce the computation time, we introduce two heuristic solution methods. The first heuristic uses the same formulation as the exact solution method, however, the number of beds needed is approximated by a linear function. The second heuristic uses a restricted version of the exact solution method within a local search approach. Hereby, the local search is used to determine the assignment of clinical departments to clusters and the exact method is used to determine the assignment of clusters to wards. The computational results show that the computation time for the two heuristic solution methods is much shorter than the computation time of the exact solution method. However, the heuristic which approximates the number of beds overestimates this number for some cases, and thus, might claim that no feasible solution exists. But when the number of beds needed is determined exactly, it might be the case that there does exist a feasible solution. Therefore,

we believe that the heuristic which uses a restricted version of the exact solution method within a local search approach is the best solution method for this problem.

Concluding, this dissertation gives several examples, where uncertainties included in the modeling process lead to complex models. To still be able to solve the problems in a reasonable amount of time, we either approximate some part of the model or use heuristics to solve the problem. Both approaches lead to suboptimal solutions, but nevertheless, the developed approaches form a good first step in solving these complex problems.

Samenvatting

Door de vergrijzing en toenemende kosten in de gezondheidszorg worden ziekenhuizen gedwongen hun processen efficiënter in te richten, zodat met minder capaciteit hetzelfde of een groter aantal patiënten kan worden geholpen. Het gaat dan bijvoorbeeld om de capaciteit van ambulances, operatiekamers en bedden. Een complicerend aspect in de gezondheidszorg is de grote mate van onzekerheid, zoals de aankomst van spoedpatiënten, de ligduur en het te volgen zorgpad. Door het zorgvuldig plannen van de processen tijdens het zorgpad van de patiënt kan de efficiëntie in de gezondheidszorg worden verbeterd. Hierbij moet op de volgende drie aandachtspunten worden gelet: (1) minimaliseren van de wachttijd, (2) creëren van robuuste planningen en (3) rekening houden met de aankomst van spoedpatiënten. Deze drie aandachtspunten zijn niet altijd even belangrijk in de in dit proefschrift bestudeerde problemen en kunnen soms zelfs conflicterend zijn. Om toch een goede afweging tussen deze (conflicterende) doelstellingen te maken, kan er een representatief gewicht aan elke doelstelling worden toegekend.

Dit proefschrift bestaat uit vier delen. Het eerste deel vormt de inleiding van het proefschrift en bestaat uit hoofdstuk 1 en 2. Hoofdstuk 1 bespreekt de uitdagingen die zich voordoen bij het efficiënter inrichten van processen in ziekenhuizen. Het vakgebied combinatorische optimalisatie en de oplosmethoden gebruikt in dit proefschrift worden geïntroduceerd in hoofdstuk 2.

In deel II tot en met IV worden praktische problemen vanuit verschillende ziekenhuizen in Nederland besproken, namelijk het HagaZiekenhuis, de Isala Klinieken en het Erasmus Medisch Centrum. Het gaat in alle gevallen om planningsvraagstukken, maar deze betreffen verschillende planningsniveaus. Samen vormen de beschouwde problemen een mogelijk zorgpad van een chirurgische spoedpatiënt.

Deel II van dit proefschrift richt zich op ambulance planning. Ambulance planning kan worden onderverdeeld in planning op het strategische, tactische en operationele planningsniveau. In hoofdstuk 3 worden oplosmethoden geïntroduceerd voor het bepalen van de ambulance standplaatsen op het strategische niveau en het toewijzen van ambulances aan standplaatsen op het tactische niveau. Daarnaast worden een aantal formuleringen voor deze problemen gepresenteerd en verschillende oplossingsstrategieën voor het gelijktijdig en afzonderlijk oplossen van beide problemen besproken. De ontwikkelde methoden zijn generiek opgezet zodat verschillende typen dekkingsvoorwaarden kunnen worden opgenomen. Hierdoor kunnen de methoden worden toegepast in acute zorgdiensten van meerdere landen. De ontwikkelde methoden zijn getest op data die gebaseerd is op de situatie in Nederland. De resultaten laten zien dat de methode die beide problemen

afzonderlijk oplost beter presteert als wordt gekeken naar het minimaliseren van het aantal standplaatsen. De methode die beide problemen gelijktijdig oplost is daarentegen beter in het minimaliseren van het aantal ambulances. Daarnaast is het met de laatstgenoemde methode makkelijker om een goede afweging te maken tussen het aantal standplaatsen en ambulances door verschillende gewichten aan beide doelen toe te kennen. Een nadeel van deze methode is dat de rekentijd exponentieel toeneemt met de grootte van het probleem terwijl de rekentijd van de methode die beide problemen afzonderlijk oplost lineair stijgt.

Deel III beschouwt planningsvraagstukken met betrekking tot de operatiekamer (OK). Hoofdstuk 4 richt zich op het minimaliseren van de wachttijd van spoedpatiënten aangezien dit het risico op complicaties vermindert. Uitgaande van de situatie dat er geen spoed-OK is, wordt een spoedoperatie gestart zodra één van de electieve operaties is afgelopen. De eindtijden van electieve operaties worden aangeduid als 'break-in-moments' (BIMs). Door deze eindtijden te spreiden over de dag kan de wachttijd van spoedoperaties worden verminderd. Om dit te bereiken, moeten de operaties per OK zo worden gepland dat de maximale tijd tussen twee BIMs wordt geminimaliseerd. Verschillende heuristieken en exacte oplosmethoden worden geïntroduceerd om dit nieuwe planningsprobleem op te lossen. De resulterende planning kan echter verstoord worden door het inplannen van spoedoperaties of doordat operaties korter of langer duren dan gepland. Hierdoor veranderen de eindtijden van de operaties en dus de BIMs, waardoor ook de maximale tijd tussen twee BIMs verandert. Met behulp van een simulatiestudie wordt het effect van deze veranderingen en de robuustheid van de planning bepaald. De resultaten van deze simulatiestudie laten zien dat de wachttijd voor spoedoperaties met 10% verlaagd kan worden in vergelijking met de huidige planning van het Erasmus Medisch Centrum.

Hoofdstuk 5 van deel III richt zich op het herplannen van operaties gedurende de dag. Door de aankomst van spoedoperaties en de variërende duur van operaties kan de operatieplanning worden verstoord. Dit kan leiden tot ongewenste situaties voor patiënten, verpleegafdelingen en andere afdelingen in het ziekenhuis. Om deze ongewenste situaties te voorkomen zal de OK-planning moeten worden aangepast als er een verstoring plaatsvindt. Het ontwikkelde decision support system (DSS) ondersteunt de OK-manager in deze beslissing door de drie beste aangepaste planningen te presenteren. Het DSS houdt rekening met de voorkeuren van alle stakeholders en beschouwt alleen de OK-planningen die voldoen aan alle voorwaarden gesteld door het ziekenhuis. De beslisregels gebruikt in het DSS zijn gebaseerd op een uitgebreide analyse van het beschouwde probleem. De eerste stap van deze analyse is het formuleren van een Integer Linear Program (ILP) met als doel de afwijking van de voorkeuren van de stakeholders te minimaliseren. Door dit ILP toe te passen op instanties van de Isala Klinieken, bleek dat voornamelijk (1) operaties werden verschoven of (2) een pauze tussen twee operaties werd gepland. Door deze beslisregels toe te passen in het DSS worden er in de Isala Klinieken minder operaties geannuleerd en wordt er meer aan de wensen van de verpleegafdelingen voldaan. Om deze verbeteringen te compenseren wordt

wel de werkdruk op andere afdelingen verhoogd. Het ontwikkelde DSS kan ook gebruikt worden om de kwaliteit van een voorgestelde OK-planning te bepalen.

Deel IV van dit proefschrift richt zich op de planning van verpleegafdelingen. In hoofdstuk 6 worden oplosmethoden gepresenteerd die de OK-planning zo aanpassen dat de bedbezetting op de verpleegafdelingen wordt gespreid zodat het aantal benodigde bedden wordt geminimaliseerd. In deze context is een OK-planning gedefinieerd als een toewijzing van blokken operaties aan beschikbare dagen in de planningshorizon waarbij rekening wordt gehouden met verschillende voorwaarden. Het bepalen van de bedbezetting is complex en vereist de convolutie van discrete kansverdelingen door de stochastische ligduur van de patiënten. Om het bepalen van de bedbezetting te vereenvoudigen worden twee aanpakken beschouwd. De eerste aanpak vermindert de complexiteit door het toepassen van een local search heuristiek die in elke stap de complexe berekening in zijn geheel uitvoert. De tweede aanpak vermindert de complexiteit door de bedbezetting te benaderen waardoor het probleem kan worden opgelost met behulp van een ILP. Beide oplosmethoden zijn getest op data van het HagaZiekenhuis in Nederland. Daarnaast is een aantal scenario's voor het HagaZiekenhuis geëvalueerd. De resultaten laten zien dat de aanpak die de bedbezetting benadert de beste resultaten geeft voor de situatie in het HagaZiekenhuis. Door het toepassen van deze aanpak kan het aantal bedden worden verminderd met bijna 20%.

Hoofdstuk 7 laat zien dat het clusteren van specialismen gunstig kan zijn wanneer het aantal beschikbare bedden beperkt is. Echter, niet alle specialismen kunnen worden geclusterd, bijvoorbeeld omdat chirurgische patiënten een infectie kunnen oplopen van niet-chirurgische patiënten. Daarnaast moeten patiënten van één specialisme niet worden verspreid over het hele ziekenhuis omdat hierdoor visite lopen gecompliceerder wordt. Het aantal bedden per afdeling is in de beschouwde situatie gegeven. De vraag is hoe de specialismen moeten worden geclusterd en toegewezen moeten worden aan verpleegafdelingen zodat het aantal beschikbare bedden voldoende is om een vooraf gespecificeerde weigeringskans te garanderen. Eerst wordt een exacte formulering van het probleem gegeven zodat de optimale oplossing kan worden bepaald. De rekeningtijd voor deze methode is dermate lang dat deze niet kan worden toegepast in de praktijk. Daarom zijn twee heuristieken ontwikkeld die de rekentijd verkorten. De eerste heuristiek gebruikt dezelfde formulering als de exacte oplosmethode maar benadert het aantal benodigde bedden met een lineaire functie. De tweede heuristiek gebruikt een beperkte versie van de exacte oplosmethode gecombineerd met een local search methode. Deze heuristiek gebruikt de local search methode om specialismen te clusteren en de exacte oplosmethode om de geclusterde specialismen toe te wijzen aan de verpleegafdelingen. De resultaten laten zien dat de rekentijd voor de heuristieken veel korter is dan de rekentijd voor de exacte oplosmethode. De eerste heuristiek vindt echter in sommige gevallen onterecht geen toelaatbare oplossing doordat het aantal benodigde bedden wordt overschat. Daarom is de tweede heuristiek de beste oplosmethode voor dit probleem.

Dit proefschrift geeft een aantal voorbeelden waarbij de aanwezigheid van on-

zekerheden leidt tot complexe modellen. Het exact oplossen van deze problemen kost door de complexiteit teveel rekentijd. Om de rekentijd te reduceren kunnen deze problemen worden vereenvoudigd door heuristieken toe te passen of een deel van het model te benaderen. Beide aanpakken resulteren in suboptimale oplossingen, maar desondanks zijn de ontwikkelde methoden een goede eerste stap om deze complexe problemen op te lossen.

About the author

Theresia van Essen was born in Dordrecht, the Netherlands, on July 27, 1985. She obtained her VWO diploma at the Lambert Franckens College in Elburg in 2003. She started studying Mathematics at the Leiden University in 2003 with a minor Music at the Royal Conservatoire The Hague. In 2004, she continued her study at the Delft University of Technology, where she enrolled in the second year of the bachelor program Applied Mathematics. In 2007, she obtained her Bachelor's of Science degree in Applied Mathematics with a thesis on pattern separation. She graduated in 2009 at the same university with a cum laude Master's of Science degree in Applied Mathematics with a thesis on the hub location problem carried out at ORTEC in Gouda.

After finishing her master studies, Theresia joined the department Applied Mathematics of the University of Twente for a Ph.D. program with the Discrete Mathematics and Mathematical Programming group. Additionally, she conducted research at HagaZiekenhuis in The Hague. In 2012, she visited the Karlsruhe University of Technology in Germany for a period of three months. Her Ph.D. research culminates with this dissertation.

Publications

J.T. van Essen, J.L. Hurink, S. Nickel, and M. Reuter. Models for ambulance planning on the strategic and the tactical level. *Beta Working Paper WP-434, Beta Research School for Operations Management and Logistics, Eindhoven,* 2013.
(Basis for Chapter 3)

J.T. van Essen, J.L. Hurink, E.W. Hans, and A. Oversberg. Minimizing the waiting time for emergency surgery. *Operations Research for Health Care,* 1 (2-3): 34 – 44, 2012.
(Basis for Chapter 4)

J.T. van Essen, J.L. Hurink, W. Hartholt, and B.J. van den Akker. Decision support system for the operating room rescheduling problem. *Health Care Management Science,* 15 (4): 355 – 372, 2012.
(Basis for Chapter 5)

J.T. van Essen, J.L. Hurink, W. Hartholt, and B.J. van den Akker. Rescheduling in the OR: a decision support system keeps stakeholders and OR-manager happy. *(E)Hospital,* 15 (4): 2013.
(Basis for Chapter 5)

J.T. van Essen, J.M. Bosch, E.W. Hans, M. van Houdenhoven, and J.L. Hurink. Reducing the number of required beds by rearranging the OR-schedule. *To appear in OR Spectrum.*
(Basis for Chapter 6)

J.T. van Essen, M. van Houdenhoven, and J.L. Hurink. Clustering clinical departments for wards to achieve a prespecified blocking probability. *Beta Working Paper WP-407, Beta Research School for Operations Management and Logistics, Eindhoven,* 2013.
(Basis for Chapter 7)

P.J.H. Hulshof, R.J. Boucherie, J.T. van Essen, E.W. Hans, J.L. Hurink, N. Kortbeek, N. Litvak, P.T. Vanberkel, E. van der Veen, B. Veltman, I.M.H. Vliegen, and M.E. Zonderland. ORchestra: an online reference database of OR/MS literature in health care. *Health Care Management Science,* 14 (4): 383 – 384, 2011.

Due to an aging population and increased healthcare costs, hospitals are forced to use their resources more efficiently. Practitioners in hospitals often have the idea that this will reduce the quality of care. On the contrary, one may argue that improving the efficiency should also improve the quality of care as, e.g., unnecessary waiting time is reduced or even eliminated. However, improving the efficiency in hospitals is challenging as many uncertainties have to be taken into account. In this dissertation, various solution methods that deal with these uncertainties and aim to minimize waiting times, create robust schedules, and account for the arrival of emergency patients are discussed.

ISBN 978-90-365-0725-7

HagaZiekenhuis van Den Haag

Beta
Research School for Operations Management and Logistics

UNIVERSITY OF TWENTE.
Faculty of Electrical Engineering, Mathematics and Computer Science
Department of Applied Mathematics
Discrete Mathematics and Mathematical Programming Group
Center for Healthcare Operations Improvement and Research