KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Operational decision support models and algorithms for hospital admission planning and scheduling

Wim Vancroonenburg

Supervisors:
Prof. dr. P. De Causmaecker
Prof. dr. F. C. R. Spieksma
Prof. dr. ir. G. Vanden Berghe

Dissertation presented in partial fulfilment of the requirements for the degree of Doctor of Engineering Science (PhD): Computer Science

January 2015

Operational decision support models and algorithms for hospital admission planning and scheduling

Wim VANCROONENBURG

Examination committee:
Prof. dr. ir. J. Berlamont, chair
Prof. dr. P. De Causmaecker, supervisor
Prof. dr. F. C. R. Spieksma, supervisor
Prof. dr. ir. G. Vanden Berghe, supervisor
Prof. dr. J. Beliën
Prof. dr. ir. D. Cattrysse
Prof. dr. B. Demoen
Prof. dr. S. Nickel
(Karlsruhe Institute of Technology, Germany)

Dissertation presented in partial fulfilment of the requirements for the degree of Doctor of Engineering Science (PhD): Computer Science

January 2015

Uitgegeven in eigen beheer, Wim Vancroonenburg, Celestijnenlaan 200A box 2402, B-3001 Leuven (Belgium)

ISBN 978-94-6018-955-5

D/2015/7515/11

Preface

Pursuing a PhD has been one of the greatest challenges I have undertaken so far. Fortunately enough, I was not alone in this venture and I owe thanks to many people that helped me get to this point. More than that: it has been a privilege to work under the auspices of three very talented researchers and well respected supervisors. Their expertise and their patience shaped the PhD dissertation now lying before you, into what it is. Therefore, a sincere "Thank you" to Patrick De Causmaecker, Frits Spieksma and Greet Vanden Berghe, for the guidance, the support and the feedback throughout my PhD.

None of the other supervisors will hold it against me that I especially mention Greet Vanden Berghe. We met for the first time in 2008 during my master studies in industrial sciences: electronics/ICT at KAHO Sint-Lieven (now KU Leuven, Technology Campus Ghent), as she supervised my master thesis. Already then she provided me with excellent guidance and helped me to achieve a nice result. Collaborating with her contributed to the build-up of my confidence and when later on she invited me to join the CODeS research group, to pursue a PhD, I was easily convinced. As the subject of my work started to take form and the direction of health care operations was taken, she brought me into contact with several hospitals that would follow-up my work. She provided me with the environment and the network of people to bring this work to a good end. For all that and much more: thank you, Greet.

The advisory committee and examination committee played important roles during the preparation of this PhD. I was always stressed to present my work and my findings to you as I saw it as an examination. However, time and again it proved to be an opportunity to interact and to improve my research. Thank you Jeroen Beliën, Dirk Cattrysse, Bart Demoen and Stefan Nickel, for accepting to partake in these committees. Thank you Jean Berlamont for chairing the examination committee.

Hospital operations are a complex matter. Even more so, not always do theory

and practice intertwine. The hospitals AZ Alma in Eeklo, AZ Sint-Lucas in Gent, and UZ Leuven, the university hospital of Leuven, provided me with essential insights into the admission processes, along with indispensable feedback. The provision of data has been invaluable. Therefore, thank you Dirk Bernard, Johan De Baere, Fritz Defloor, Rudy Maertens, André Orban, Marc Van der Weyde, Annabell Verhaegen, from AZ Alma; Tine van Langenhove, from AZ Sint-Lucas; Christian Lamote, Pierre Luysmans, Philip Monnens, Bart Smeets, Jo Vandersmissen, Nancy Vansteenkiste from UZ Leuven.

I am also very thankful to IWT, the Flemish agency for Innovation through Science and Technology, for the grant awarded to do this research. IWT provided me with a stable income for the past four years and the necessary funding to do research and to present my work abroad at numerous conferences.

Part of my research took place abroad, during a research visit to the *Politecnico di Torino*, the polytechnic university of Turin, Italy from mid October 2012 to mid December 2012. I am very grateful to Federico Della Croce for inviting me to collaborate with him and his colleagues Fabio Salassa, Andrea Grosso and Michele Garraffa. I would like to thank all of them for showing me the Italian hospitality, the country's amazing culture and not to forget, great coffee.

Working for the CODeS research group meant in the first place being part of a young and enthusiastic group of researchers, passionate about combinatorial optimization and computer science. Luckily enough, it was not all computer science: video and board game nights, table soccer matches, team building activities and countless lunches and coffee breaks with vivid discussions, they were all part of the game. In no particular order: thank you Tony, Jannes, Pieter, Joris K., Jan C., Tulio, David, Thomas S., Eline, Thomas V.d.B. and Evert-Jan. Our colleagues of the MSEC research group were never far away and often joined in on the discussions. Thank you Vincent, Jan V., Faysal, Koen D., Laurens and Michiel. Quite a few people left us in the last five years to pursue careers elsewhere. I hope they are all doing well. Thank you Peter, Katja, Joris M., Jorn, Koen V., Wouter, Tim, Burak, Mustafa, Murat, Mike and Sam. Special thanks go to my colleague Erik Van Achter, for reviewing and correcting my English texts.

Last but not least, I would like to thank my family and friends for their continued support. My parents have always been there for me, believing that I could do it even when I was much more pessimistic. They were especially worried if "I wasn't working too hard" when I was writing this dissertation. Fortunately enough, my wife Sarah made sure that I was getting a stable vitamin supply (fresh fruit juices every day) to keep me going. Thank you Sarah for standing by me and supporting me in every possible way for all these years. Especially during the past half year when I spent far too little time with you when I was

writing this dissertation. Often she would find me still working on my text well past midnight, when I should have been getting a good night's sleep beside her. However, this may have been a good training for what's to come, as we are expecting our first child to start keeping us awake from late April 2015 onwards.

Wim Vancroonenburg,
Ghent, January 2015.

Abstract

The present dissertation focuses on developing operational decision support models and algorithms for hospital admission planning and scheduling. The aim is to increase efficient usage of key hospital resources by supporting human planners at hospital admission offices with automated tools for their daily and weekly decision making. Three planning processes concerned with admission scheduling of patients are considered: assignment of admitted patients to hospital rooms, determination of admission dates for elective surgical patients, and scheduling surgical cases in operating rooms.

The planning process of assigning patients to hospital rooms and wards is the subject of two studies. Firstly, a reactive and an anticipatory decision support model are presented for daily decision making on patient-to-room assignments. It is shown that the anticipatory model is better than the reactive model under various conditions. The reactive model can be seen as an idealized version of current hospital practices, implying that current decision making can be improved and efficient usage of a diverse set of hospital rooms can be increased. Secondly, the Red-Blue transportation problem (Red-Blue TP) is introduced as an abstraction of the patient-to-room assignment problem. A complexity and computational study on the Red-Blue TP provide insights into the difficulty of patient-to-room assignment planning under a *gender separation policy*.

The third and fourth studies concentrate on the admission scheduling process and operating theatre scheduling process for surgical patients. For the admission scheduling process, the aim is to support human planners in determining *when* patients should be admitted such that expected operating theatre costs and patient waiting time are minimized, while considering limited bed availability. A stochastic optimization model and a heuristic algorithm are presented, that serve as the basis for developing admission scheduling strategies. It is shown that, when given sufficient planning flexibility, stochastic optimization models may improve on deterministic decision models by considering the variance in bed usage and operating theatre usage. However, this improved performance is at the expense of patient friendliness, quality of care and throughput.

Finally, for the operating theatre scheduling process, a general and flexible decision support model is presented capturing many considerations encountered in practice. It aims to support human planners in determining a schedule for performing surgical cases in the operating theatre while considering a variety of resources by means of a generalized resource model. Additionally, the model's objectives are to increase throughput and the efficient usage of the operating theatre and its resources. A heuristic algorithm is developed to solve the model which scales well with problem size.

Beknopte samenvatting

Deze verhandeling beschrijft de ontwikkeling van beslissingsondersteunende modellen en algoritmen voor opnameplanning in ziekenhuizen op een operationeel beslissingsniveau. Het beoogt de inzetting van de meest cruciale ziekenhuismiddelen efficiënter te maken, door menselijke opnameplanners in hun dagelijkse taken te ondersteunen met geautomatiseerde tools. In dit werk worden drie processen beschouwd: het toewijzen van opgenomen patiënten aan ziekenhuiskamers en afdelingen, het toekennen van opnamedata aan electieve chirurgische patiënten, en het inplannen van chirurgische ingrepen in het operatiekwartier.

Het proces waarin patiënten worden toegewezen aan ziekenhuiskamers wordt beschouwd in de eerste twee studies van deze verhandeling. In de eerste studie worden een reactief en een anticipatief beslissingsondersteunend model opgesteld voor het dagelijks toewijzen van patiënten aan ziekenhuiskamers. Een computationele studie toont aan dat het anticipatieve model beter presteert dan het reactieve, en dat dit blijft gelden in verschillende omstandigheden. Dit impliceert dat het huidige plannen verbeterd kan worden, aangezien het reactieve model als een geïdealiseerde versie van de huidige planningsaanpak gezien kan worden. In een tweede studie wordt het *Rood-Blauw transport probleem* geïntroduceerd als abstractie van het kamertoewijzingsprobleem. Aan de hand van een studie van de complexiteit, alsook aan de hand van een computationele studie, worden inzichten verworven betreffende de moeilijkheid van kamertoewijzingsproblemen waarbij een *genderrestrictie* van toepassing is.

De laatste twee studies van de verhandeling hebben betrekking op het bepalen van opnamedata en het inplannen van chirurgische ingrepen voor chirurgische patiënten. In de opnameplanningsstudie is het doel om opnameplanners te ondersteunen in het bepalen van opnamedata voor electieve, chirurgische patiënten zodanig dat kosten in het operatiekwartier alsook patiëntwachttijden kunnen worden geminimaliseerd. Hierbij moet echter ook rekening worden gehouden met de beschikbare bedcapaciteit. Een stochastisch optimalisatiemodel en een heuristisch algoritme worden beschreven voor dit probleem, die verder dienen voor de ontwikkeling van verschillende opnameplanningsaanpakken. In een rekenexperiment wordt aangetoond dat, indien de aanpak voldoende beslissingsvrijheid heeft, een stochastisch model een meerwaarde biedt omdat het de variantie in bedbezetting en in bezetting van het operatiekwartier beter kan inschatten. Echter, deze meerwaarde komt ten koste van de patiënt, die op korte termijn kan opgeroepen worden voor de opname en mogelijk langer moet wachten op de opname.

In de laatste studie van de verhandeling komt het planningsproces van het operatiekwartier aan bod. Hiervoor wordt een algemeen en flexibel beslissingsmodel ontwikkeld dat vele in de praktijk voorkomende aspecten in acht neemt. Ook hier is het doel planners van het opnamekwartier te ondersteunen bij hun dagelijkse taak van het inplannen van chirurgische ingrepen, waarbij rekening moet worden gehouden met de beschikbaarheid van verschillende actoren en middelen. Om de diversiteit van deze middelen te vatten wordt een veralgemeend *resource* model voorgesteld. Het beslissingsmodel beoogt de reguliere benutting van het operatiekwartier te verhogen en de middelen van het operatiekwartier zo goed mogelijk in te zetten. Hiertoe wordt een heuristisch algoritme ontwikkeld dat goed schaalt met de probleemgrootte.

Abbreviations

DRG	Diagnosis related group
ED	Emergency department
GDP	Gross domestic product
ICU	Intensive care unit
IP	Integer programming
LAHC	Late Acceptance Hill Climbing
LOS	Length of stay
MILP	Mixed integer linear programming
MSS	Master surgical schedule
OR	Operating room
OR/MS	Operations research and management science
OT	Operating theatre
PA	Patient-to-room assignment
PACU	Post anaesthetic care unit
SAA	Sample average approximation
SCSP	Surgical case scheduling problem
TP	Transportation problem

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and scope

Health care systems are facing incredible challenges. Over the past decades, globally rising expenditures on health care have forced governments to re-evaluate health care funding. In Belgium for example, health care spending accounted for 8.5% of the gross domestic product (GDP) in 2002; in 2012, it already accounted for 10.9% and it is still expected to increase [58]. To reduce public spending on health care, budgetary pressure on hospitals has increased significantly. At the same time, demand for hospital services has increased, due to e.g. population ageing [24]. Hospitals are expected to perform more with less resources. Hospital managers are thus constantly looking into new ways to increase efficiency, whilst maintaining a high level of care.

Operations research and management science (OR/MS) as fields of study have identified ample opportunities in health care systems to support decision making and to improve efficiency. An increasing volume of literature on the subject bears testimony of these opportunities. A recent overview of the literature on OR/MS applied to health care services is given by Hulshof et al. [40]. Over 400 studies are cited, and the majority have only been published in the last decade. Hulshof et al. present a taxonomy that classifies studies on two axes. On the horizontal axis, studies are classified according to the health care service(s) they apply to. Services that have been considered are: ambulatory care services, emergency care services, surgical care services, inpatient care services, home care services and residential care services. On the vertical axis, studies are classified according to the level of detail and the time horizon of the decision making, ranging from strategic decision making that impacts over a long term, to tactical decision making over a medium term, to offline and online operational decision making on the short resp. real-time term.

Clearly, the application possibilities for OR/MS techniques in health care and hospitals are broad. It is far beyond the scope of the present dissertation to cover all. Rather, this dissertation zooms in on one fundamental process that impacts hospital operations: the admission of patients. It is the very process of admitting, and treating, patients that is an essential service of any hospital and that generates revenue, costs and profit. Considerable resources, infrastructure and staff are dedicated to providing this service and their efficiency is key in order to keep operating costs under control. A great deal of planning, and the consideration of several resources, is necessary in order to achieve this goal.

Within the taxonomy of Hulshof et al. [40], the admission process can generally be located in the inpatient care services category, as by definition an inpatient is an admitted or *hospitalized* (staying for minimum 1 night) patient. The primary resources that are under consideration are hospital wards, care units, their bed capacity and personnel. However, the admission process also touches other service categories such as surgical care services, as surgery is a major causal factor for patient admission.
With respect to the vertical decision levels, admission planning decisions are made on all levels. At the highest, strategic decision and planning level, capacity (e.g. infrastructure, beds, personnel) is allocated to different medical services and disciplines based on case-mix decision making and projections of demand and possible profit. At the tactical decision level, the organization of the capacity allotted to different medical services is handled for the mid term. Well studied examples are the decision on personnel rosters that meet staffing requirements for the upcoming month(s) and the development of admission control schemes that aim for timely access for patient groups and maximal occupancy, whilst cancellations and misplacements are minimized. Finally, at the operational decision level, decisions are made on what individual patient admissions are planned, and what tasks personnel must perform. More important, with respect to admissions, the decision is made on *when* individual patients are admitted, but also *where* (in which ward and room).

Figure 1.1 shows a general scheme of the admission process and patient flow. The admission process generally considers three types of admission flows: planned or *elective* admissions, unplanned patients entering through the emergency department (ED), and finally outpatients requiring admission. The first major flow is the ‘normal’ flow of patients, having previously visited a physician or specialist and who were called in for admission or had a booked admission to undergo a treatment. The second major flow is the emergency flow of patients, who entered the hospital through the ED and, after triage and

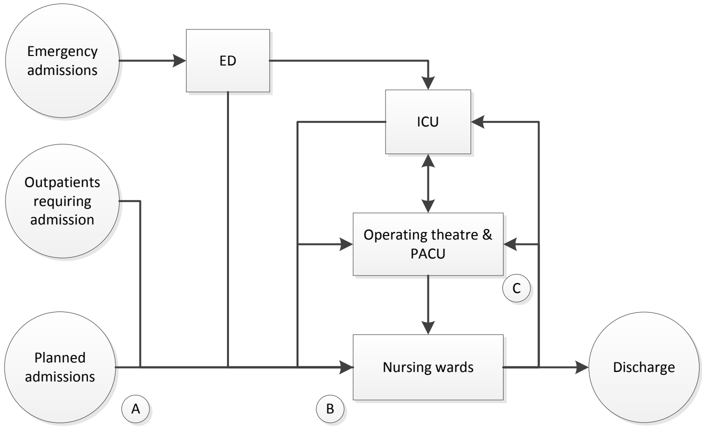


Figure 1.1: Scheme of the admission process and patient flow applicable to patients requiring surgery. (A),(B),(C): points considered for operational decision support. A: admission planning for elective surgical patients; B: Patient-to-room assignment planning; C: Operating theatre scheduling.

application of primary care, were determined to be admitted to the hospital. The third, rather minor, flow is that of outpatients. Outpatients are those patients that have undergone treatment in the hospital's ambulatory facilities (i.e. for treatments not requiring an overnight stay). Due to a longer than expected recovery process, or complications, these patients are sometimes admitted for further recovery or treatment (and thus become *in*patients).

Upon admission, any of these patient types requires a bed to stay in and is admitted to a room. Typically, this will be in a nursing ward, but for severe cases an intensive care unit (ICU) is used. During the hospitalization, a patient may be transferred between several wards and care units depending on availability and medical requirements, before the patient ultimately exits the hospital (by discharge, referral, or regrettably, by death).
Figure 1.1 also shows the interaction with the operating theatre (OT)[1]. As noted by van Oostrum et al. [74], as high as 60% of hospital admissions may be surgery related. The OT is a key resource in any hospital and is considered to be the largest cost and revenue center [13]. Therefore, one cannot disregard the OT when looking at the admission process.

[1]In this dissertation the term 'operating theatre' is used to denote the general unit where surgeries are performed, comprising several operating rooms and supporting facilities.

This dissertation focuses on the development of computational models and algorithms that may provide decision support for the admission process at the operational decision level. Three points of decision support are considered, shown in Figure 1.1 by points (A), (B) and (C). In (A), the operational scheduling of hospital admissions is considered, determining *when* elective patients should be admitted. Two key capacity-constrained resources are considered: the available number of beds in nursing wards, and the available operating theatre capacity. In (B), the operational planning of patient-to-room assignment is considered. Rather than assigning patients to wards, this planning process considers to what rooms patients are admitted, in order to maximize patient care and comfort. Finally in (C), the operational scheduling of the operating theatre is considered. This scheduling process is more fine grained, determining daily schedules for the operating theatre such that a given set of surgeries can be performed within availabilities of the OT and all related resources (surgical team, equipment).

The main research questions for these applications are: *Can these processes be planned more efficiently using decision support models and algorithms?* and *How does this impact the quality-of-care?*

1.2 Structure of the dissertation

The dissertation builds on the results of four studies that were made on the processes described in the previous section. The sequence in which they are presented follows the chronology in which these studies were performed.

Chapter 2 starts with the discussion on the patient-to-room assignment (PA) planning process. The PA process has had the focus at the onset of this doctoral study due to earlier work of the author in which the PA process is studied in an offline decision making setting. In the present study, it is studied in an *online* decision making setting. The main discussion is on the comparison of two planning approaches: a reactive approach that only considers new admissions on a day-by-day basis, and an anticipative approach that also considers future, but planned, arrivals. A computational study shows that the latter is clearly beneficial, and computationally tractable.

Chapter 3 focuses on the computational complexity of the PA problem, and in particular on the gender separation policy. At the onset of the PA problem study, the complexity status of the problem was still open. This chapter abstracts the PA problem to a transportation problem in which a partitioning of the supply nodes into two sets is considered, and between which exclusionary constraints are imposed. This abstract problem serves as the basis for a complexity study, the formulation of different mixed integer programming models and a study of approximation algorithms for a maximization variant. A key result is the resolution of the PA complexity status, being NP-Hard.

Chapter 4 discusses a robust admission scheduling model for elective surgical patients in an online scheduling setting. The model considers a chance constrained bed usage formulation to constrain the risk of bed shortages in surgical wards. A sample average approximation of this stochastic model is presented, that is solved by a local search approach. This approximation model is used in different admission scheduling strategies, that are compared with each other and with other scheduling strategies in a computational study.

Chapter 5 provides an insight into operating theatre scheduling. A rich multi-day surgical case scheduling problem is presented that considers generalized resource constraints and desiderata from the surgical staff. The aim is to schedule as many surgical cases in as few operating theatres as possible, within regular operating theatre opening hours and under limited resource availability. The problem description and data were provided in the context of an IWT research project with a software company developing a software solution for operating theatre planning and scheduling. A heuristic algorithm is presented to solve this rich problem formulation and it is tested on a set of real-world data. The results are compared to surgical plans made by manual planners. The final goal of this study is the inclusion of the algorithm into the company's software application.

Finally, Chapter 6 summarizes the main contributions of this dissertation and gives some directions for future research.

Chapter 2

Patient-to-room assignment planning in an online setting

Every day, many patients are admitted to hospitals to undergo medical diagnosing, testing and treatment. Their admission was either planned beforehand or was the result of a request for admission from the emergency department or from ambulatory care services. Either way, upon admission every patient requires a room and a bed to stay in until treated, and to recover after treatment. The task of bed managers at the admission office, or possible nursing staff at nursing wards, is to determine to which wards and to which rooms and beds these patients are admitted. Different criteria are considered and it is difficult to find the best arrangement.

This chapter discusses the patient-to-room assignment planning process in such a daily planning setting. To this end, an extension of the patient assignment (PA) problem formulation as defined by [21] and further formalized by [15] is proposed, for which two online mixed integer linear programming-models are developed. The first model targets the optimal assignment for newly arrived patients, whereas the second also considers future, but planned, arrivals. Both models are compared on an existing set of benchmark instances from the PA planning problem, which serves as the basic problem setting. These instances are then extended with additional parameters to study the effect of uncertainty on the patients' length of stay, as well as the effect of the percentage of emergency patients. The results show that the second model provides better results under all conditions, while still being computationally tractable. Moreover, the results show that pro-actively transferring patients from one room to another is not necessarily beneficial.

This chapter is a minor adaptation of *W. Vancroonenburg, P. De Causmaecker, and G. Vanden Berghe. A study of decision support models for online patient-to-room assignment planning. Annals of Operations Research, 2013. doi: 10.1007/s10479-013-1478-1 . Available online.* [78]

2.1 Introduction

Rooms and beds belong to the critical assets of any hospital since they account for a considerable part of a hospital's infrastructure. A large amount of financial resources are invested in equipping them with medical apparatus to facilitate patient care. Moreover, they also represent the place where most patients will spend the largest part of their stay, as they recover from surgery or treatment, wait for examinations to take place, etc. In order to improve their comfort, patients are offered a choice between double bed rooms, single bed rooms, luxury rooms with private showers, and other conveniences. As a result, many hospitals provide a large variety of rooms in terms of capacity, medical apparatus and amenities. Assigning patients to rooms and meeting their medical requirements and personal preferences can therefore be challenging, necessitating an efficient plan for making such assignments.

Bed managers at admission offices aim at finding assignments that strike a balance between patients' preferences and comfort, and patients' clinical conditions and the resulting required room facilities. However, the availability of rooms and equipment needs to be considered, as well as hospital policies, organisation and standards, complicating decision making. A lack of overview on occupied beds and the uncertainty on how long patients will stay in the hospital, further complicate the matter.
For example, patients are preferably admitted to a room and bed in a ward that is managed by the medical discipline the patient's pathology belongs to. However, due to high occupancy at this unit a bed may not be available, thus requiring the patient to be admitted elsewhere. In addition, many hospitals employ a gender separation policy, prohibiting male and female patients to be assigned to the same room at the same time. Therefore, a male patient may still be admitted in a different unit even if there is availability, if all available rooms are already partially occupied by female patients. On the other hand, these female patients may be transferred and grouped in a room, thus freeing a room for admitting a male patient. However, patient transfers also require nursing staff to move patients (and their belongings) and adds to their workload. Such trade-offs occur frequently and must be dealt with swiftly.

Demeester et al. [21] defined and studied the patient assignment (PA) problem in the context just described. They consider a set of patients, each with individual characteristics, who arrive at a hospital over a certain period of time. The hospital comprises a set of rooms, each with a given capacity and characteristics. The problem is to find an effective assignment of patients to rooms, satisfying room capacity restrictions. Moreover, a perceived cost is associated with each patient-to-room assignment relating to the *appropriateness* of this assignment (which may also depend on the assignment of other patients). The objective is to minimize the total cost of these assignments. This chapter focuses on the PA problem.

2.1.1 Related work

The PA problem considered by Demeester et al. [21] comprises an assignment problem that occurs at the operational level of hospital admission offices. It assumes that patients have already been attributed an admission date, a decision that is made as part of either an advance scheduling process (see Chapters 4 and 5) during operational surgery scheduling, or an appointment/treatment scheduling process when no surgery is required. Demeester et al. introduced the PA problem – only recently – to the academic community as a challenging combinatorial optimization problem. In a follow up paper, Bilgin et al. [7] presented a new hyper-heuristic algorithm to the PA problem, providing new benchmark instances and reporting test results. Vancroonenburg et al. [79] showed that the PA problem is NP-hard. This work is covered in Chapter 3.

The PA problem was also studied by Ceschia and Schaerf [15], who developed a Simulated Annealing algorithm that improves on the best known results by Bilgin et al. [7] for the benchmark instances. Lower bounds for these instances are provided as well. Interestingly, Ceschia and Schaerf argue that the problem definition only assumes patients that are planned in advance (elective patients), and that it does not capture the dynamics of uncertainty on patient arrivals and departures. An extension to the problem definition is proposed where patient admission and discharge dates are revealed a few days before they occur (denoted as the *forecast level*). To this end, Ceschia and Schaerf developed a dynamic version of their algorithm that can be used for day-to-day scheduling. The performance of this algorithm is analysed under an increasingly larger forecast level. Ceschia and Schaerf [16] continued their efforts in developing a dynamic version of the PA problem formulation, also considering registration dates, the possibility of delaying patients, and minimizing the risk of room overcrowding.
Range et al. [65] presented a column generation approach to the PA problem. They were able to find tighter lower bounds for the benchmark instances than

those presented by Ceschia and Schaerf [15]. In addition, their approach is able to find new best known solutions.

Other related studies have also studied patient-to-room assignment problems, though not specifically the formulation by Demeester et al. [21]. Mazier et al. [55] presented a real-time patient assignment method, assigning both elective patients and incoming patients from the emergency department. Their aim was to reduce waiting time of emergency patients to admission to a ward bed, without disrupting inpatient stays too much. A simulation study shows that they reached their goal, reducing the number of patient transfers and having a low waiting time. Furthermore, they compared a to-room assignment formulation with a to-ward assignment formulation, showing that the latter is computationally more tractable.
Thomas et al. [72] presented a comprehensive decision support system for hospital-wide patient-to-bed assignments. This support system considers all aspects mentioned above, as well as staffing considerations (specifically, nurse-to-patient ratios). Interestingly, it does not consider patients' length of stay. Thomas et al. showed that their support system was effective in automatically providing bed assignments to more than 90% of the bed requests, and a reduction in waiting time was observed for emergency department patients to be assigned a ward bed. Validation with bed management staff confirmed that the automated bed assignments were satisfactory.

With respect to patient admission scheduling (i.e. also considering determination of the admission date), problem settings and decision support systems considering bed/room assignment issues, such as gender separation policies, room preferences, have received – recently – increasing attention in literature. Bachouch et al. [2] presented a hospital bed management problem where patient admissions are scheduled, considering no-mixed gender rooms, isolation of contagious patients in single rooms or alone in double rooms, incompatibilities between pathologies, etc. Bachouch et al. developed a mixed integer programming formulation of the problem and applied both free and commercially available solvers.
Schmidt et al. [67] presented an admission scheduling approach that considers the availability of room preferences as well as the distinction between rooms for male and female patients. The problem is studied in a dynamic setting, with consideration of adaptable length of stay estimates. An integer programming model of the problem is presented and an exact approach is compared with heuristic strategies in a simulation study.

In a more general consideration of related work, the PA problem where patient transfers are not allowed, is related to the interval scheduling problem: patients can be represented by fixed length intervals (i.e. jobs with fixed start and end time) that need to be assigned to a machine (a room) for 'processing'. The

PA problem comprises required jobs and non-identical machines with different capacities, the goal being to find a minimum-cost schedule subject to side-constraints. In a dynamic context, it constitutes an *online* interval scheduling problem with uncertainty on the interval lengths. In the case where patients are allowed to be transferred from one room to another, the problem can be seen as an interval scheduling problem which permits pre-emption of jobs. Kolen et al. [46] provide a review on the subject of (online) interval scheduling problems. However, a critical difference with the interval scheduling problem is that the PA problem also includes costs which are directly related to sets of patients being assigned to the same room (gender conflicts, see Section 2.2), in contrast to the costs related to a single patient-room assignment. This already makes the problem hard (see Chapter 3), for a single time-unit instance (which effectively drops the notion of intervals).

2.1.2 Contribution

The present study was motivated by the work in [15]. This study complements the work by Ceschia and Schaerf [16], who discuss the PA problem in a dynamic context. We similarly define a new extension to the PA problem in a dynamic context. To this end, registration dates for each patient are added to the problem definition signalling a patient's possible future arrival time. This contrasts with the approach of defining an absolute forecast level [15], which assumes that all patient arrivals within the forecast level are known. Such an approach does not allow accurate modelling of emergency patients.

Moreover, the present study makes a more general assumption on the patients' length of stay (LOS). Only the availability of an estimate on each patients' LOS is assumed, which in practice is often the case (either by historical data, or the physician's estimate). However, this requires to make adjustments to previous decisions when patients outstay their estimated LOS (and thus room-assignment collisions occur). This contrasts with the work by Ceschia and Schaerf [16], who model this issue as a static overcrowd risk that should be avoided. The effect of replanning patients on the solution quality is not considered. The question remains how to address this, should it occur. In our study, special care is taken to accommodate this specific decision process.

This dynamic version of the problem is modelled and solved using Integer Linear Programming (ILP). The performance of this approach is discussed and the effect of the percentage of emergency cases and the accuracy of the LOS estimate is studied. It is shown that taking into account information on future, but registered, arrivals allows for improved decision making, even in the presence of emergency arrivals and inaccurate LOS estimates.

2.2 Problem definition

2.2.1 Patient-to-room assignment in a static context

The PA problem as described by Demeester et al. [21] considers a set of patients P that each need to be assigned to one of a set of hospital rooms R over a certain planning horizon $H = \{1, \ldots, T\}$ (typically over 1-2 weeks). Each room $r \in R$ has a given capacity, denoted by $c(r)$. Each patient $p \in P$ is attributed an arrival time $a(p)$ and a departure time $dd(p)$, with the time interval $H_p = \{t \in H : a(p) \leq t < dd(p)\}$ representing the patient's stay in the hospital. The length of the patient's stay, $dd(p) - a(p) = |H_p|$, is denoted as $los(p)$.

The problem is to find an assignment $\sigma : P \times H \mapsto R$ of patients to rooms, for each time unit of their stay, that minimizes a certain cost $w(\sigma)$ related to these assignments. This cost $w(\sigma)$ consists of three parts:

- Total patient/room assignment cost: each patient/room combination can be judged based on different criteria:
  - Is the unit in which the room is located suited for treating the patient's pathology (i.e. does it have the right specialism)?
  - Does the patient's age violate any age restriction imposed in certain units (e.g. paediatric and geriatric units).
  - Is the room suitably equipped for treating the patient's pathology?
  - Does the room meet the patient's room preferences (e.g. is it a single room, if requested)
  - Does the patient need to be quarantined in a single room?
  - ...

  Demeester et al. define several of these considerations that each are penalized (with specific weights for each criterion) in the objective function.

  Ceschia and Schaerf [15] note that all of these penalties can be 'compiled' in a penalty-matrix $c(p, r)$ (lower is better). This approach was adopted in this study. Therefore, the goal is to minimize the sum of these assignment costs, where each individual cost is weighted by the LOS of a patient.

$$\text{Min } w_1(\sigma) = \sum_{p \in P} \sum_{t \in H_p} los(p) \cdot c(p, \sigma(p, t)) \tag{2.1}$$

- The total number of *gender conflicts*: one penalty that cannot be modelled by the penalty matrix $c(p, r)$ is the gender separation constraint. This constraint states that in some rooms (which are denoted 'dependent' by Demeester et al., denoted $R_{DEP} \subseteq R$), either male or female patients may be admitted; however, no male and female patient should be admitted to the same room at the same time. Therefore, the goal is to avoid such gender conflicts:

$$\text{Min } w_2(\sigma) = \sum_{r \in R_{DEP}} \sum_{t=1}^{T} Conflict_{\sigma rt} \tag{2.2}$$

with:

$$\begin{aligned} Conflict_{\sigma rt} = \min(\quad & |p \in P_{\sigma rt} : p \text{ is } male|, \\ & |p \in P_{\sigma rt} : p \text{ is } female|) \end{aligned} \tag{2.3}$$

denoting the minimum number of patients that must be moved in order to solve the gender conflict, and:

$$P_{\sigma rt} = \{p \in P : a(p) \leq t < dd(p), \sigma(p, t) = r\} \tag{2.4}$$

denoting the set of patients assigned to room r at time t by assignment σ.

- The total number of patient *transfers*: The PA problem formulation also allows patients to be transferred from one room r to another room r' during their stay, if this is beneficial or necessary. Examples where this may be the case are: upgrading a female patient if all available rooms are partially occupied by male patients, downgrading a patient from a single bed room to a double bed room when another patient needs isolation, etc. Transfers also cause grievances for patients and additional workload for the nursing staff. Thus, transfers from one room r to another r' should also be minimized:

$$\text{Min } w_3(\sigma) = \sum_{p \in P} \sum_{t \in H_p \setminus \{a(p)\}} Transfer_{\sigma pt} \tag{2.5}$$

with

$$Transfer_{\sigma pt} = \begin{cases} 1 & \text{if } \sigma(p, t-1) \neq \sigma(p, t), \\ 0 & \text{otherwise.} \end{cases} \tag{2.6}$$

The complete objective can then be expressed as follows:

$$\text{Min } w(\sigma) = w_1(\sigma) + w_G \cdot w_2(\sigma) + w_{Tr} \cdot w_3(\sigma) \tag{2.7}$$

with w_G, w_{Tr} weights denoting the relative importance of gender conflicts and transfers. Finally, the assignment σ should respect the room capacities at all times, i.e. :

$$\forall t = 1, \ldots, T,\ r \in R : \quad |P_{\sigma rt}| \leq c(r) \tag{2.8}$$

2.2.2 Extension to an online, dynamic context

In practice, the arrivals and departures of patients are gradually revealed over the planning horizon. The present contribution therefore extends the problem definition to account for these dynamics. Each patient p is attributed a *registration* date $r(p)$, at which point the patient becomes known to the system, and an *expected* departure date, $ed(p)$, which is an estimate of the patient's departure date. The actual departure date of the patient, $dd(p)$, however remains hidden until it has passed.

The dynamic problem definition requires a new problem to be solved at each $t' \in H$ (i.e. at the start of each day) where the following information is available:

- $P_{t'}$: the set of patients with $r(p) = t'$, i.e. the patients that are registered at time t'. At this point, only $a(p)$ and $ed(p)$ are known for each patient $p \in P_{t'}$, $dd(p)$ remains hidden.
- $DP_{t'}$: the set of patients with $dd(p) = t'$, i.e. the patients that leave the hospital at time t'.

as well as the information in $P_1, P_2, \ldots, P_{t'-1}$ and $DP_1, DP_2, \ldots, DP_{t'-1}$. Let $A_{t'}$ denote the set of patients that arrived at t', i.e. :

$$A_{t'} = \{p \in P : a(p) = t'\} \tag{2.9}$$

The goal of the problem is to find at each time t' an assignment

$$\sigma_{t'} : \left(\bigcup_{i=1}^{t'} A_i \right) \times \{1, \ldots, t'\} \mapsto R \tag{2.10}$$

that maps each *arrived* patient p (i.e. all p for which $a(p) \leq t'$) to a hospital room r, for each time unit of their individual stay up till t'. Obviously, the assignment $\sigma_{t'}$ should still respect room capacity at all times. The following condition should also hold:

$$\forall p \in \left(\bigcup_{i=1}^{t'-1} A_i \right), t \in [a(p), t'-1[: \quad \sigma_{t'}(p,t) = \sigma_{t'-1}(p,t) \tag{2.11}$$

i.e. the new assignment σ'_t should respect decisions made at times $1, \dots, t' - 1$.

The assignment σ_T denotes the solution at the end of the planning horizon. It contains all the patients' assignments within that period. The solution quality can be assessed by computing $w(\sigma_T)$, which is again the sum of patient-to-room assignment costs, gender conflicts over the entire planning horizon, and finally patient transfer penalties. It is interesting to compare this value with the quality obtained for the static variant of the problem, which assumes that each patient's departure date is fixed in advance. Any lower bound for the static version is a lower bound for the dynamic problem, and thus is an indication of what can be achieved when all information is known *a priori*.

2.3 Optimization models

We developed two models for the dynamic PA planning problem that correspond with the decision that must be made at time t', that is: give a new assignment of patients to rooms considering the current situation.

The first model is modelled after current practice, namely the assignment decision is made shortly before patient arrival and only current room availability is considered. The model tries to find the optimal assignment for the patients who arrived at t'. Moreover, it uses the estimate of the newly arrived patients' LOS. If any admitted patient stays longer than expected (i.e. $ed(p) \leq t'$ and $p \notin DP_1, DP_2, \dots DP_{t'}$), it is assumed that the patient stays at least one time unit longer.

The second model builds on the previous model by also considering all registered patients at each t', therefore anticipating future occupancy and room demand. This approach can be seen as an online algorithm with *lookahead*: the model is aware of future arrivals. Dunke [23] studies online optimization with lookahead, providing a modelling framework and presenting both theoretical and experimental results on different case studies. Dunke concludes that an overall positive effect can be observed when lookahead is introduced for online algorithms. However, this effect is also dependent on key problem characteristics such as: the allowance to take advantage of lookahead; the degree of freedom for the online algorithm; the possibility of making bad decisions; the performance quality gap between online and offline (post-hoc) algorithms. Such characteristics will be studied in the computational study in Section 2.4.

Both models are implemented as Mixed Integer Linear Programming models. They are described in Sections 2.3.1 and 2.3.2. In order to simplify the description, the following notation will be used:

- $\mathbf{P_{t'}} = \bigcup_{i=1}^{t'} P_i \setminus \bigcup_{i=1}^{t'} DP_i$, the set of all registered patients, that have not yet left the hospital, up to (and including) t',
- $\mathbf{A_{t'}} = \bigcup_{i=1}^{t'} A_i \setminus \bigcup_{i=1}^{t'} DP_i$, the set of all arrived patients, that have not yet left the hospital, up to (and including) t',
- superscript M, F, restrict a set of patients P to either *males* or *females* respectively,
- $elos(p) = \max(ed(p), t'+1) - \max(a(p), t')$, the remaining, expected length of stay of patient p as it is known at decision time t'. If the patient's stay has exceeded his or her expected departure date $ed(p)$, he or she is expected to stay at least one time unit longer.
- $\mathbf{AP_{tt'}} = \{p \in \mathbf{A_{t'}} : t' \leq t < \max(ed(p), t'+1)\}$, the set of *arrived* patients that are *expected* to be present at time t ($t \geq t'$),
- $\mathbf{PP_{tt'}} = \{p \in \mathbf{P_{t'}} : t' \leq t < \max(ed(p), t'+1)\}$, the set of *registered* patients that are *expected* to be present at time t ($t \geq t'$).
- $Transfer_{t'pr} = \begin{cases} 1 & \text{if } \sigma_{t'-1}(p) \neq r \text{ (i.e. patient } p \text{ is moved on day } t'), \\ 0 & \text{otherwise.} \end{cases}$
- $MaxClique_{t'}(P)$, the set of subsets of P, whose intervals, starting from t', form maximal cliques in the corresponding interval graph. We refer to Section 2.3.1 for more information.

2.3.1 Model 1: reactive assignments

The decision variables are defined as follows:

$$x_{p,r} = \begin{cases} 1 & \text{if patient } p \text{ is assigned to room } r, \\ 0 & \text{otherwise.} \end{cases} \tag{2.12}$$

$$v_{r,t} = \text{the number of } \textit{gender} \text{ conflicts in room } r \text{ at time } t \tag{2.13}$$

$$y_{r,t} = \begin{cases} 1 & \text{if the number of } \textit{male} \text{ patients assigned to room } r \\ & \text{at time } t \text{ is larger than or equal to the number of} \\ & \textit{female} \text{ patients,} \\ 0 & \text{otherwise.} \end{cases} \tag{2.14}$$

The optimization problem is then modelled as follows:

$$\text{Min} \sum_{p \in \mathbf{A_{t'}}} \sum_{r \in R} (elos(p) \cdot c(p,r) + w_{Tr} \cdot Transfer_{t'pr}) \cdot x_{p,r}$$

$$+ \sum_{r \in R_{DEP}} \sum_{t=t'}^{T} w_G \cdot v_{r,t} \tag{2.15}$$

subject to:

$$\sum_{r \in R} x_{p,r} = 1 \qquad \forall p \in \mathbf{A_{t'}} \tag{2.16}$$

$$\sum_{p \in P_c} x_{p,r} \leq c(r) \qquad \forall r \in R, P_c \in MaxCliques_{t'}(\mathbf{A_{t'}}) \tag{2.17}$$

$$\sum_{p \in \mathbf{AP^M_{tt'}}} x_{p,r} \geq v_{r,t} \qquad \forall r \in R_{DEP}, t = t', \ldots, T \tag{2.18}$$

$$\sum_{p \in \mathbf{AP^F_{tt'}}} x_{p,r} \geq v_{r,t} \qquad \forall r \in R_{DEP}, t = t', \ldots, T \tag{2.19}$$

$$\sum_{p \in \mathbf{AP^M_{tt'}}} x_{p,r} \leq v_{r,t} + c(r) \cdot y_{r,t} \qquad \forall r \in R_{DEP}, t = t', \ldots, T \tag{2.20}$$

$$\sum_{p \in \mathbf{AP^F_{tt'}}} x_{p,r} \leq v_{r,t} + c(r) \cdot (1 - y_{r,t}) \qquad \forall r \in R_{DEP}, t = t', \ldots, T \tag{2.21}$$

$$x_{p,r} \in \{0,1\} \qquad \forall p \in \mathbf{A_{t'}}, r \in R$$

$$v_{r,t} \geq 0 \qquad \forall r \in R_{DEP}, t = t', \ldots, T$$

$$y_{r,t} \in \{0,1\} \qquad \forall r \in R_{DEP}, t = t', \ldots, T$$

The model describes an assignment problem minimizing the expected cost of the newly arrived patients (Expression 2.15). Constraint (2.16) specifies that each arrived patient has to be assigned to a room, while constraint (2.17) expresses that room capacity should be respected for all maximal cliques in the interval graph corresponding to $\mathbf{A_{t'}}$ (see below for more information). Constraints (2.18), (2.19), (2.20), and (2.21) relate the variables $v_{r,t}$ and $y_{r,t}$, forcing $v_{r,t}$ to take on the expected value of the minimum number of either males or females in room r at time t.

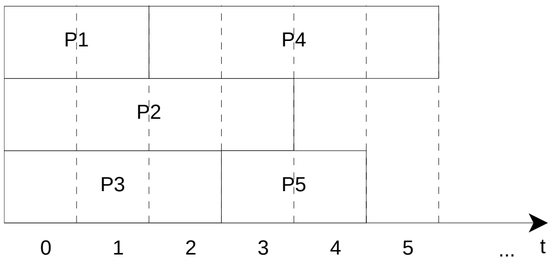


Figure 2.1: An example showing five patient intervals (patient stays) for which the capacity should be enforced. Implementing capacity constraints for this situation can be done more efficiently by generating a constraint for the max-clique of the corresponding interval graph, rather than imposing a capacity constraint for $t = 1, 2, \ldots, 6$.

Using maximal cliques for room capacity

At any given time t, the room capacity constraint needs to be respected. A straightforward way to implement this constraint is to add the following expression to the model:

$$\sum_{p \in \mathbf{AP_{tt'}}} x_{p,r} \leq c(r) \quad \forall r \in R, t = t', \ldots, T \tag{2.22}$$

However, this is a fairly inefficient way of implementing this constraint as the following example shows. Consider the patient intervals (patient stays) shown in Figure 2.1. For any given room r with capacity $c(r)$, the following constraints would be imposed using formulation (2.22):

$$x_{1,r} + x_{2,r} + x_{3,r} \leq c(r) \qquad t = 0 \tag{2.23}$$

$$x_{1,r} + x_{2,r} + x_{3,r} \leq c(r) \qquad t = 1 \tag{2.24}$$

$$x_{2,r} + x_{3,r} + x_{4,r} \leq c(r) \qquad t = 2 \tag{2.25}$$

$$x_{2,r} + x_{4,r} + x_{5,r} \leq c(r) \qquad t = 3 \tag{2.26}$$

$$x_{4,r} + x_{5,r} \leq c(r) \qquad t = 4 \tag{2.27}$$

$$x_{4,r} \leq c(r) \qquad t = 5 \tag{2.28}$$

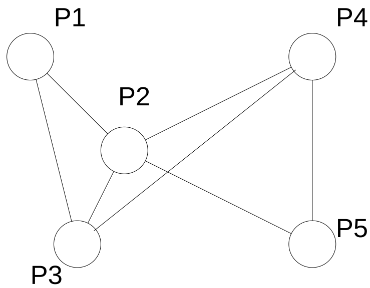


Figure 2.2: The interval graph corresponding to the intervals in Figure 2.1.

It is clear that (2.23)-(2.24) are identical, and (2.27) and (2.28) are already implied by (2.26). A more efficient way of implementing this constraint would be as follows:

1. Construct an *interval graph* based on the intervals from a given subset of patients P'. An interval graph is a graph $G(V, E)$ where each vertex $v \in V$ corresponds to an interval (in this case the interval of a patient $p \in P'$). The edge set is given by $E = \{\{v_1, v_2\} | v_1, v_2 \in V \wedge v_1 \text{ overlaps } v_2\}$, i.e. there is an edge between two vertices v_1, v_2 if their corresponding intervals overlap. Figure 2.2 shows the interval graph corresponding to the example in Figure 2.1.

2. Enumerate all *maximal cliques* from this interval graph. A clique C is a subset of vertices ($C \subseteq V$), such that $\forall v_i, v_j \in C \Rightarrow \{v_i, v_j\} \in E$, i.e. a clique is a subset of nodes which are pairwise directly connected. The maximal cliques of G are all cliques C for which $C \cup \{v'\}$, with $v' \in V \backslash C$, does not form a clique. That is, maximal cliques[1] are cliques of maximal cardinality and can not be expanded by adding any node from V not in C. The maximal cliques of the interval graph in Figure 2.2 are $\{p_1, p_2, p_3\}$,$\{p_2, p_3, p_4\}$ and $\{p_2, p_4, p_5\}$.

3. For each room $r \in R$, and each maximal clique P_c (corresponding to a clique C in the interval graph), construct a capacity constraint:

$$\sum_{p \in P_c} x_{p,r} \leq c(r) \tag{2.29}$$

[1]Note that the notion of a maximal clique differs from a *maximum* clique. A *maximum* clique is the largest cardinality clique that can be found in a graph.

In the example, the following constraints are constructed:

$$x_{1,r} + x_{2,r} + x_{3,r} \leq c(r) \qquad \text{for clique } \{p_1, p_2, p_3\} \tag{2.30}$$

$$x_{2,r} + x_{3,r} + x_{4,r} \leq c(r) \qquad \text{for clique } \{p_2, p_3, p_4\} \tag{2.31}$$

$$x_{2,r} + x_{4,r} + x_{5,r} \leq c(r) \qquad \text{for clique } \{p_2, p_4, p_5\} \tag{2.32}$$

which are identical to resp. (2.23), (2.25) and (2.26)

Consider these maximal cliques to correspond with the maximal subsets of patients from P' present in the hospital at any given time. Thus, to enforce that room capacity is respected, it is sufficient to enforce that for each of these subsets no more than $c(r)$ patients may be assigned to a room r, $\forall r \in R$. As these cliques are maximal, any smaller clique of these patients that may be present at a later time, is already implied by the constraint for the maximal clique.

Enumerating the maximal cliques of an interval graph can be done in polynomial time. Krishnamoorthy et al. [47] describe such an algorithm, which is the algorithm that was implemented for this work. We truncate the start of the intervals, corresponding to patients in P', such that the decision time t' is the smallest time considered. That is:

$$\forall p \in P' : H'_p = \{t \in H : \max\left(a(p), t'\right) \leq t < dd(p)\} \tag{2.33}$$

In this way, the capacity constraint is checked only from t' onwards, and thus ensures that the resulting constraints properly allow transfers of patients.

The maximal clique model requires fewer constraints than the former one. We compared the two formulations (with and without the maximal clique version of the capacity constraint) of this model on a set of instances, based on instance 5 of the test set (refer to Section 2.4.1 for more details on the experimental setup). In this setting, the total number of constraints generated by the model (for solving the first decision problem, at $t' = 0$) was reduced from 5939 to 4885 (averaged over 300 models generated), a reduction by 17.7 %.

2.3.2 Model 2: anticipatory assignments

The second model defines the same decision variables as Model 1, but it differs in the set of patients for which they are defined. The $x_{p,r}$ variables are defined for all arrived patients $\mathbf{A_{t'}}$ in the first model, whereas they are defined for all *registered* patients $\mathbf{P_{t'}}$ in the new model. Another difference is that patients can be assigned to a *dummy* room, denoted as $\perp$. Only registered patients who have not arrived ($p \in \mathbf{P_{t'}} \backslash \mathbf{A_{t'}}$) are allowed in this dummy room, so as to ensure feasibility of the model under an expected, future undercapacity. These assignments are attributed a high cost $c(p, \perp)$ in such a way that the model gives priority to a real assignment for each future arrival. Finally, the model also does not require that the assignment for registered, not-arrived patients takes on an integer value. Therefore, the model takes into account a *lower bound* on the assignment cost for these patients, which speeds up calculations while still allowing for an informed decision on the current assignments.

The model is defined as follows:

$$\text{Min} \sum_{p \in \mathbf{P_{t'}}} \sum_{r \in R \cup \perp} (elos(p) \cdot c(p,r) + w_{Tr} \cdot Transfer_{t'pr}) \cdot x_{p,r}$$

$$+ \sum_{r \in R_{DEP}} \sum_{t=t'}^{T} w_G \cdot v_{r,t} \tag{2.34}$$

subject to:

$$\sum_{r \in R \cup \perp} x_{p,r} = 1 \qquad \forall p \in \mathbf{P_{t'}} \tag{2.35}$$

$$\sum_{p \in P_c} x_{p,r} \leq c(r) \qquad \forall r \in R, P_c \in MaxCliques_{t'}(\mathbf{P_{t'}}) \tag{2.36}$$

$$\sum_{p \in \mathbf{PP^M_{tt'}}} x_{p,r} \geq v_{r,t} \qquad \forall r \in R_{DEP}, t = t', \ldots, T \tag{2.37}$$

$$\sum_{p \in \mathbf{PP^F_{tt'}}} x_{p,r} \geq v_{r,t} \qquad \forall r \in R_{DEP}, t = t', \ldots, T \tag{2.38}$$

$$\sum_{p \in \mathbf{PP^M_{tt'}}} x_{p,r} \leq v_{r,t} + c(r) \cdot y_{r,t} \qquad \forall r \in R_{DEP}, t = t', \ldots, T \tag{2.39}$$

$$\sum_{p \in \mathbf{PP^F_{tt'}}} x_{p,r} \leq v_{r,t} + c(r) \cdot (1 - y_{r,t}) \qquad \forall r \in R_{DEP}, t = t', \ldots, T \tag{2.40}$$

$$x_{p,\perp} = 0 \qquad \forall p \in \mathbf{A_{t'}} \tag{2.41}$$

$$x_{p,r} \in \{0,1\} \qquad \forall p \in \mathbf{A_{t'}}, r \in R \cup \perp$$

$$0 \leq x_{p,r} \leq 1 \qquad \forall p \in \mathbf{P_{t'}} \backslash \mathbf{A_{t'}}, r \in R \cup \perp$$

$$v_{r,t} \geq 0 \qquad \forall r \in R_{DEP}, t = t', \ldots, T$$

$$y_{r,t} \in \{0, 1\} \qquad \forall r \in R_{DEP}, t = t', \ldots, T$$

The objective of the model, expression (2.34), is again to minimize the total assignment cost, including minimizing any possible dummy assignments. Constraints (2.35) specify that each arrived and registered patient should be assigned to one room, allowing for dummy assignments for future arrivals. Constraints (2.36) - (2.40) are similar to their counterparts in Model 1, this time also considering future arrivals. Constraint (2.41) ensures that arrived patients are not assigned to dummy rooms.

2.4 Computational study

2.4.1 Experimental setup

The anticipative model distinguishes itself from the reactive model by including lookahead. However, this lookahead is not *complete*: urgent admissions and patient departures are not known beforehand. In addition, a first-come first-served admission policy is not necessarily in effect. An elective patient that registered later than another patient may still be admitted earlier than that patient. Therefore, the possibility of adverse effects (bad decisions) when planning anticipatively is real.

To thoroughly test the reactive model and the anticipative model, the following problem factors were considered:

- Accuracy of the length of stay estimate: the length of stay estimate determines how well we can estimate the assignment cost ($c(p, r) \cdot los(p)$) of patients. The worse the estimate is, the higher the chance of making bad decisions.
- Emergency versus planned cases: if more patients are admitted without an earlier registration (i.e. admissions from the emergency department), flexibility is reduced and the chance of making bad decisions is higher.
- Occupancy: if occupancy is high, flexibility of assigning patients to rooms will be reduced.

In addition, we investigated whether or not allowing patient transfers in the model has a large impact on the final solution quality of the algorithms. Therefore, two versions of both the reactive model (Model 1) and the anticipatory

instance	$\|P\|$	$\|R\|$	$\sum_{r\in R} c(r)$	avg. occupancy (%)	$\|H\|$
1	652	98	286	59.69	14
5	587	102	325	49.32	14
8	895	148	441	43.90	21
10	1575	104	308	47.76	56

Table 2.1: Problem characteristics of the instances.

model (Model 2) were tested: one version corresponds to the previously described models (see Section 2.3) that allow transfers, whereas the second version does not allow transfers. The latter implies that the second model fixes the assignment of a patient after arrival, or mathematically:

$$\forall p \in \mathbf{A_{t'-1}} \cap \mathbf{A_{t'}}, r = \sigma_{t'-1}(p) \Rightarrow x_{p,r} = 1 \tag{2.42}$$

For the purpose of experimentation, a subset of the benchmark instances for the static PA problem was used (available from the patient admission scheduling website [20]). The instances served as the basic problem setting to test these models, which were then extended to test for the above mentioned factors.

The instances were extended to the dynamic problem by adding a random registration date $r(p)$ and an expected departure date $ed(p)$ for each patient p over the planning horizon. We refer to Table 2.1 for the characteristics of these instances. The weights $c(p,r)$ are based on the decision rules (is a patient assigned to a department with the correct specialism, are their room preferences met, etc.) discussed by Demeester et al. [21] and detailed by Demeester [20]. The weights w_G and w_{Tr} are set to 5.0 and 11.0, corresponding to the weights used by Demeester et al. However, note that in the implementation/testing of the model, these weights have been multiplied by 10 in order to obtain an integer representation, rather than a decimal representation (Demeester et al. use weights with accuracy up to 1 decimal place).

The procedure for enriching the instances is as follows:

- $ed(p)$ is selected uniformly from the interval $[dd(p) - acc, dd(p) + acc]$ for each patient individually. If $ed(p) <= a(p)$, then it is set to $ed(p) = a(p)+1$. We investigated the effect of acc, i.e. the effect of the accuracy of the expected departure date estimate.

- $r(p)$ is either selected uniformly from the interval $[a(p) - T, a(p) - 1]$ for planned patients, or is set to $a(p)$ for emergency patients. We investigated the effect of the percentage (denoted em) emergency versus planned cases.

Both models (and both the transfer and non-transfer versions) were tested on all combinations of the factors *acc* (LOS estimate) and *em* (percentage emergency cases), with *acc* ranging from 0 time units (perfect estimate) to 5 time units (a poor estimate) and $em \in \{0, 0.25, 0.50, 0.75, 1.0\}$. All tests were performed on 10 randomized instances for each specified combination of the mentioned factors.

The effect of increasing occupancy was tested by randomly removing beds (uniformly selected) from the instances in order to increase the projected average occupancy. This procedure is similar to what is done in [15]. Feasibility is maintained by limiting the peak occupancy to 100% (i.e. bed capacity remains at least equal to the size of the maximum clique of the interval graph of patient intervals). For studying the effect of increasing occupancy, the lower bound on the offline problem solution was calculated for every occupancy setting since it increases as beds are removed from the instance. This lower bound is calculated by solving the linear relaxation of the MILP formulation of the instance. In the figures discussed in the following section, this lower bound is denoted as LB.

The ILP models have been implemented using CPLEX 12.4 with a free academic license.The computations were conducted on the infrastructure of the VSC - Flemish Supercomputer Center, funded by the Hercules foundation and the Flemish Government - department EWI. All experiments were performed on computers equipped with an 8 core, 2.8 GHz Xeon X5560 (Nehalem) processor, and 24 GB of ram memory, running a GNU/Linux operating system. The supporting code was implemented in Java 1.7. The CPLEX 12.4 solver was configured to use only one processing thread, enabling us to test up to 8 instances in parallel on one machine, reducing computation time. In total, 9 machines from the Flemish Supercomputer Center were used for these tests, reducing the total computation time (wall-clock time) 72-fold. The overall average computation time (over all instances/experiments) for a complete run of one instance was 4.80 minutes, with a maximum computation time of 227.15 minutes. Although powerful computer resources were used for these computer tests, the single core performance of such a machine (used for one complete run of an instance) is comparable to a high-end desktop computer available in 2009. Similar computation times can thus be expected on recent consumer hardware. All results and graphics were processed with the R software environment for statistical computing [64].

2.4.2 Results and discussion

Emergency versus planned cases, and the effect of the LOS estimate

Figure 2.3 shows the effect of an increasing percentage of emergencies on the value $w(\sigma_T)$, the value of the solution obtained at the end of the planning horizon. This has been averaged over all runs and parameter settings of the LOS estimate. The results show that the anticipatory models (Model 2) consistently outperform the reactive models (Model 1), for all problem instances: for each setting of the parameter em, the cost value $w(\sigma_T)$ is lower for the anticipatory models than for the reactive models. In the limit for increasing percentage of emergencies, the result of the anticipatory models converge to those of the reactive models. Obviously, in the case of 100% emergencies, no future arrivals can be planned and the anticipatory models reduce to the reactive models. Due to an excessive computation time, the setting $em = 0\%$ is not shown for instances 5, 8 and 10. To avoid complicating interpretation of the results, no time limit was set to ensure the models are solved to optimality at each t'.

Instance 1 (Figure 2.3, topleft), shows a clear advantage to allow transfers in the reactive model, while there is no clear advantage to allow them in the anticipatory models. For instance 5, 8 and 10, the results lie differently. Although the anticipatory models still outperform the reactive models, the reactive model that does allow transfers now clearly performs worse than the version that does not. Although seemingly counter intuitive, namely allowing more flexibility causing a worse performance, this can be explained by the dynamics of the problem. For example, consider that at time t' it might appear beneficial to transfer one or several patients in order to obtain a better overall bed assignment. Then, at time $t' + \Delta t$ some patients not known at time t' might arrive who now obtain a much poorer assignment due to the assignments made at time t. Thus, both the transfer cost at time t' is incurred, as well as the assignment cost at time $t' + \Delta t$.

Figure 2.4 shows the average number of occurrences of transfers in σ_T for an increasing percentage of emergencies, for instances 1 and 5. It is clear that transfers do occur when enabling them in the models. The anticipatory model clearly avoids transfers more than the reactive model, as it has more information on the planned arrivals.

Figure 2.5 shows the effect of the models' performance under increasingly poorer LOS estimates, for instances 1, 5, 8 and 10. The results show that the performance of both the reactive models (Model 1) and the anticipatory models (Model 2) deteriorates for an increasingly poorer LOS estimate, while the anticipatory models always outperform the reactive models. This result is

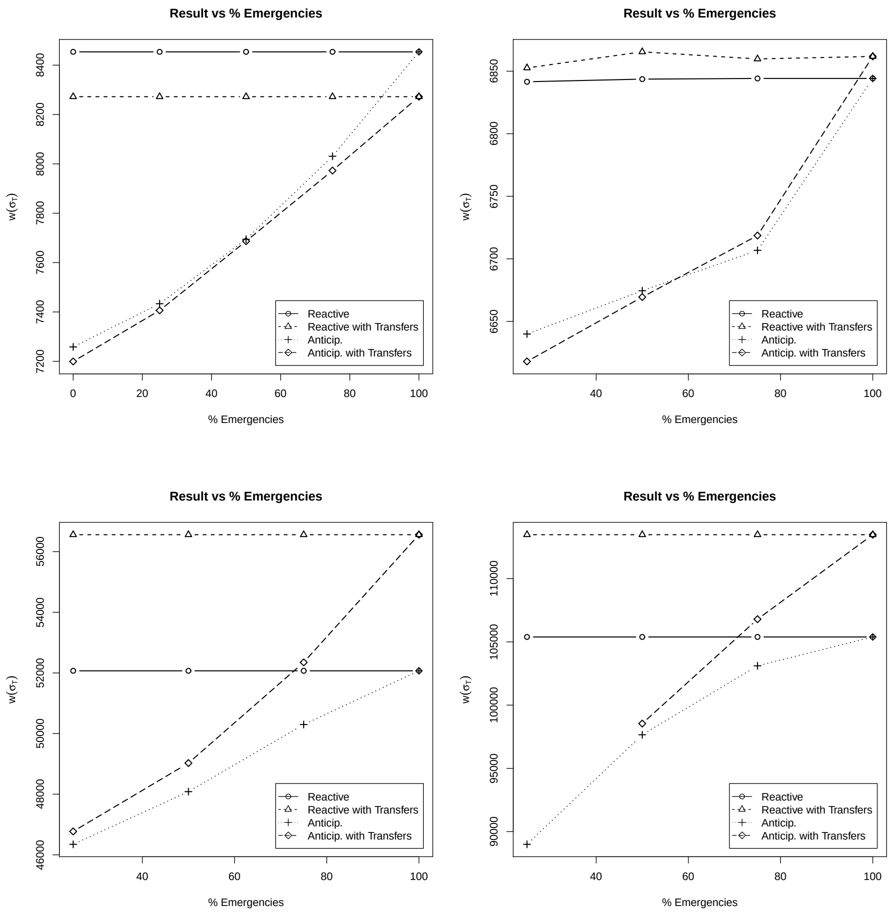


Figure 2.3: Model performance for increasing percentage of emergencies. Results are shown for instances 1 (topleft), 5 (topright), 8 (bottomleft) and 10 (bottomright).

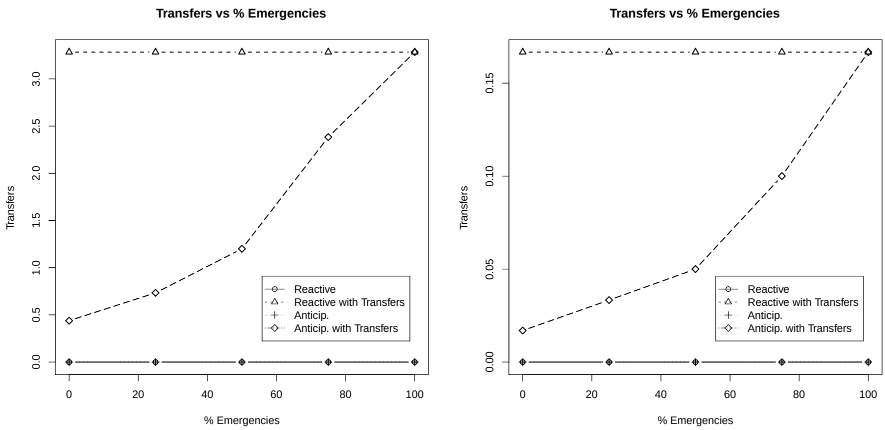


Figure 2.4: Average # transfers for increasing percentage of emergencies. Results are shown for instances 1 (left) and 5 (right).

expected, as an increasing inaccuracy of the LOS estimate causes inaccurate weighing of the patient assignments and thus generates suboptimal solutions.

Furthermore, the graphs show a more erratic behaviour, which appears unrelated to the percentage of emergencies. The reason for this behaviour is that a decision (both for the anticipatory and reactive models) may turn out good or bad when patients depart earlier or later than estimated. However, the overall trend is a decreasing performance for all models.

Another important factor that also needs to be considered is the execution time required to solve the assignment problem at each t'. Figure 2.6 shows the average execution time for the different models, with respect to the percentage of emergency arrivals. It is clear that the anticipatory models require more time to solve the problem at each t', and this difference becomes larger as the percentage of emergencies decreases. This is of course expected, because the anticipatory models require more variables as the number of elective patients increases. Furthermore, it is clear that the models which do allow patient transfers require more computation time. Not allowing patient transfers constrains the model a lot more (resulting in many variables being removed in the MIP solver presolve phase). We can also report several outliers, in terms of execution time, for the anticipatory models, where the MIP solver would take a very long time (exceeding 1 hour of computation time for solving the MIP at a specific t'). However, this always occurs at time $t' = 0$ and for $em = 0\%$, which is the initial decision problem and the case of no emergencies. In this case, a great

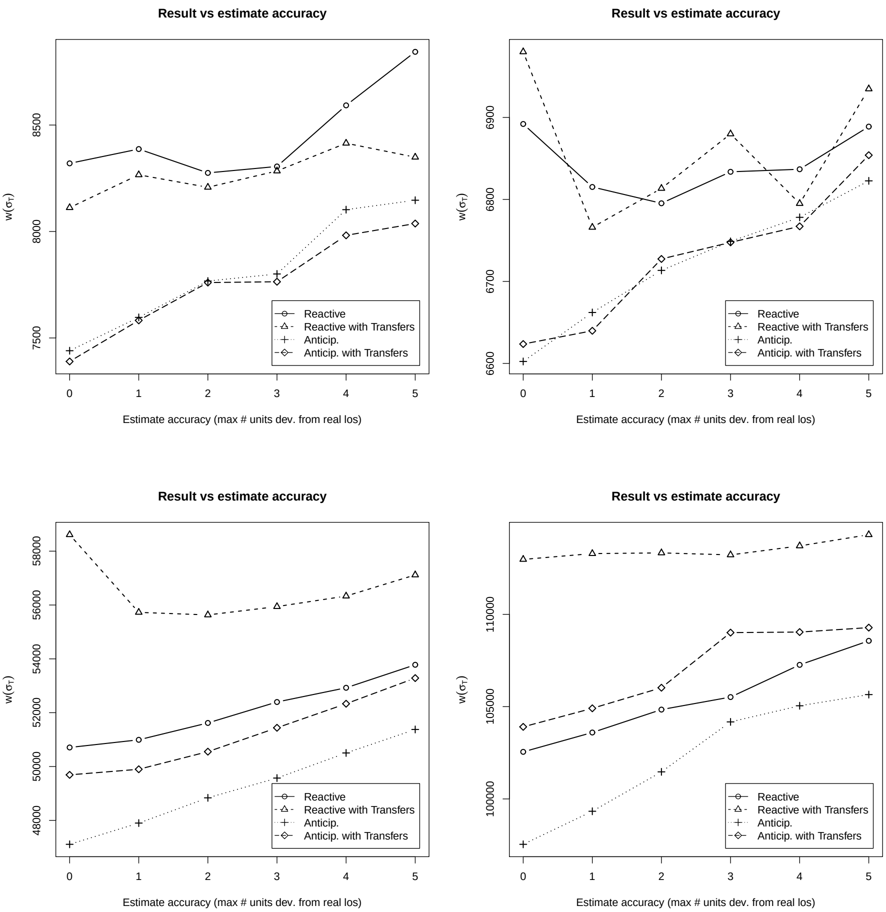


Figure 2.5: Model performance for an increasingly poorer LOS estimate. Results are shown for instances 1 (topleft), 5 (topright), 8 (bottomleft) and 10 (bottomright).

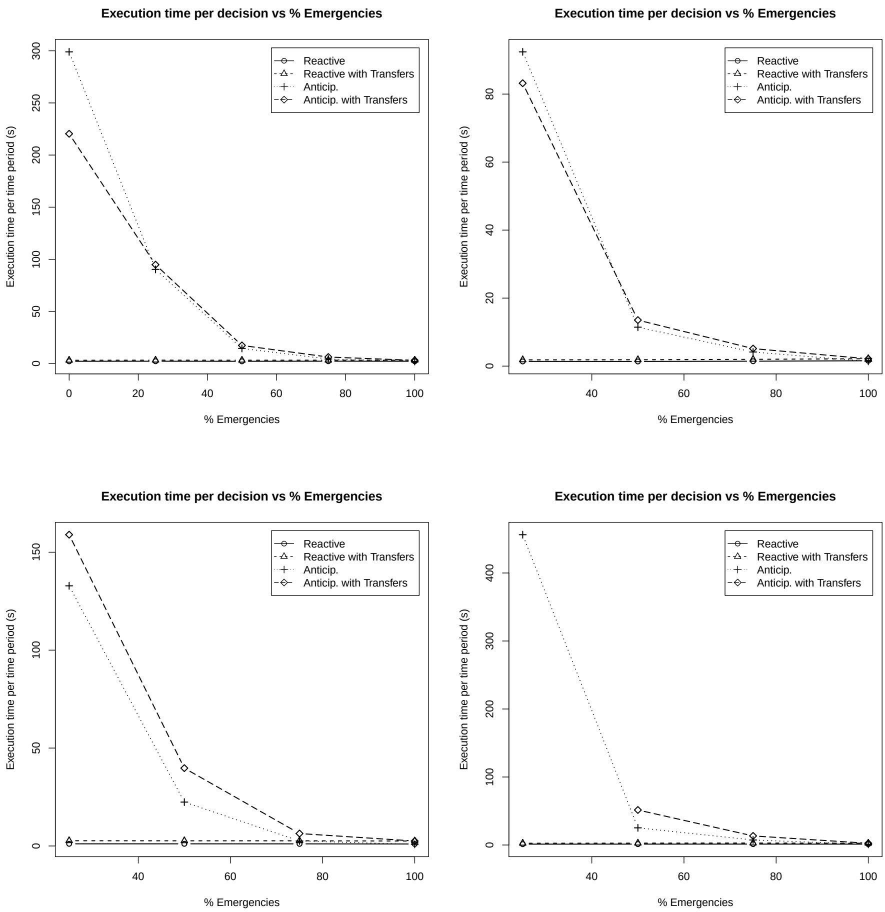


Figure 2.6: Average execution time for increasing percentage of emergencies. Results are shown for instances 1 (topleft), 5 (topright), 8 (bottomleft) and 10 (bottomright).

number of patients are taken into account (all patients registered before $t' = 0$), often more than half of the patients considered over the complete time horizon. For subsequent t', the execution time is much lower, as the MIP solver can make use of the previous solution for a *warm start* (i.e. in this case MIP solver heuristics can quickly produce a very good initial feasible incumbent, which often speeds up the branch-and-bound phase of the search). This is clearly shown in Figure 2.7, which compares the execution time with respect to the percentage of emergency arrivals, for $t' = 0$ (left) and $t' = 1$, (right).

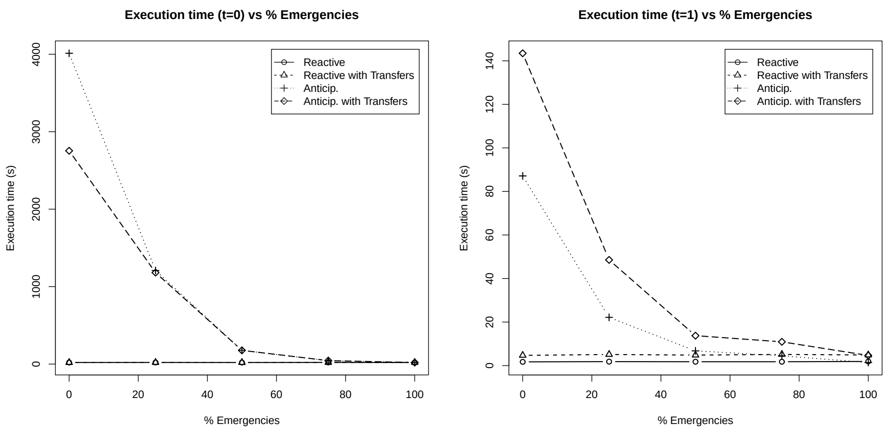


Figure 2.7: Average execution time for increasing percentage of emergencies at $t' = 0$, (left) and at $t' = 1$ (right). Results shown for instance 1.

Effect of increasing occupancy

The effect of an increasing occupancy was tested by artificially forcing a higher, average occupancy in instances 1 and 5, ranging from 59% to 77% for instance 1 and from 49% to 67% for instance 5. Both instances reach a peak occupancy of 100%. Again, all combinations of factors were tested 10 times to reduce random effects. The following results report on the averages of those 10 runs.

Figure 2.8 shows the effect of an increasing occupancy on the performance of both models, under an increasing percentage of emergencies (from left-to-right, top-to-bottom) for instance 1. It is clear that both models perform worse under an increasing occupancy. However, the lower bounds of the instances also increase as beds are removed from the instances. Thus, the relative performance of the models compared to the lower bound does not change, indicating that occupancy does not have an effect on the relative behaviour of both models.

2.5 Conclusion

The contribution of this chapter is a study of the patient assignment problem as defined by Demeester et al. [21], in a dynamic setting. The work extends the existing patient assignment problem definition from an offline setting to an *online* one. The problem formulation accounts for the dynamics of daily patient arrivals, including emergency patients, and explicitly models patients' length of stay as an *estimate*. This definition clearly maps more closely to the current practice of hospital admissions, where often only 50 percent of the patients are electively planned and patients' LOS are not known *a priori*.

Two mixed integer linear programming models were developed that model decision making when new patients arrive: one that is modelled after current practice, namely assigning patients to rooms as they arrive; and one that also accounts for planned arrivals. The first model improves on current practice by also considering the expected LOS of patients, therefore enabling proper weighing of patient assignments. Furthermore, this model is solved to optimality (with respect to the data provided), whereas in practice, hospital admission officers employ rule-of-thumb heuristics, and often do not account for the LOS of patients, leading to suboptimal solutions. The second model also accounts for future, but planned, arrivals in order to weigh patient assignments even better. It does so by including a lower bound on the future arrivals, based on a *relaxed* assignment of the corresponding patient intervals.

Experimental results showed that the second model yields a better global result than the first model, because it considers more available information on future arrivals. In addition, this model can still be solved in a reasonable amount of time using a commercial solver in most practical cases (percentage of emergencies larger than 25 percent). Experimentation with the percentage of emergency patients, poorer LOS estimates and an increasing hospital occupancy indicate that this behaviour does not change under these conditions, advocating the use of the anticipatory model over the reactive one independently of the various factors considered. Lastly, allowing patients to be transferred from one room to another is not necessarily beneficial both in terms of computation time and computational result.

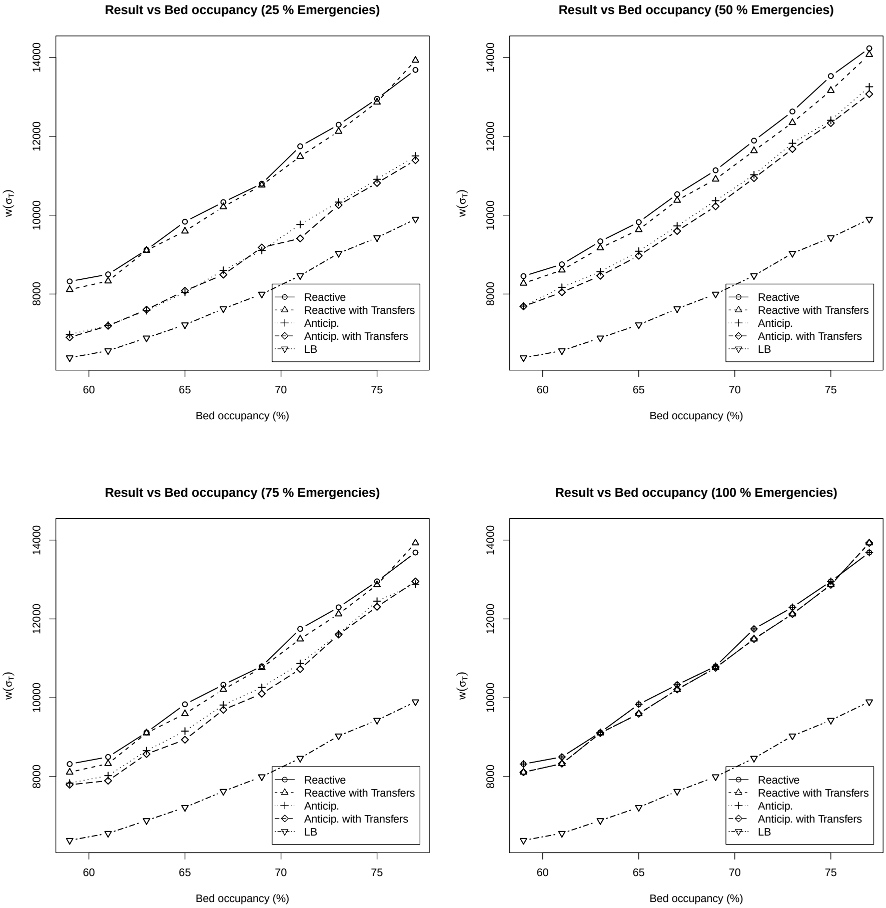


Figure 2.8: Model performance for an increasingly higher occupancy rate, under different levels of emergency vs planned patients. Results are shown for instance 1 with a perfect estimate.

Chapter 3

Computational complexity of patient-to-room assignment planning

One research question that has remained open with respect to the PA problem is whether or not the problem is difficult from a computational complexity point of view. Ceschia and Schaerf [15] showed that when no transfers costs are considered (i.e. patients can be rearranged every day) and no gender separation policy is imposed, the problem decomposes into single day sub-problems, that each reduce to an assignment problem. Therefore, in such a setting the PA problem can be solved by applying a polynomial time algorithm (e.g. the Hungarian method [48]). However, inclusion of the gender constraint in, e.g. an integer programming formulation changes the situation, resulting in a non-integer relaxation. The question then arises as to what the influence of the gender constraint is on the computational complexity of the problem.

This chapter defines the Red-Blue Transportation Problem (Red-Blue TP), a generalization of the transportation problem where supply nodes are partitioned into two sets and so-called exclusionary constraints are imposed between the two sets. The Red-Blue TP serves as an abstraction of the patient-to-room assignment problem restricted to just a single-day planning horizon, with exclusionary side constraints modelling gender separation. The problem's complexity is established, and two integer programming formulations are presented and compared. Furthermore, a maximization variant of Red-Blue TP is presented, for which we propose constant-factor approximation algorithms. Finally, a computational experiment on the performance of the integer programming formulations and the approximation algorithms is presented, that studies the problem size, the partitioning of the supply nodes, and the density of the problem.

This chapter is a minor adaptation of *W. Vancroonenburg, F. Della Croce, D. Goossens, and F.C.R. Spieksma. The Red-Blue transportation problem. European Journal of Operational Research, 237(3):814–823, 2014. doi: 10.1016/j.ejor.2014.02.055.* [79]

3.1 Introduction

Consider the well-known Transportation Problem (TP): given is a set of supply nodes S, each with supply a_i ($i \in S$), a set of demand nodes D, each with demand b_j ($j \in D$), with $\sum_{i \in S} a_i = \sum_{j \in D} b_j$, and a bipartite graph $(S \cup D, E)$, with a given cost c_{ij} for each edge $(i, j) \in E$, where E is not necessarily complete. The question is how to send the flow from supply nodes to the demand nodes such that the total flow cost is minimum. The TP is easily formulated as an integer programming problem. Define the decision variables x_{ij} as the amount of supply that node i sends to node j. The formulation is then:

$$\textbf{Min} \sum_{(i,j) \in E} c_{ij} \cdot x_{ij} \tag{3.1}$$

subject to:

$$\sum_{(i,j) \in E} x_{ij} = a_i \quad \forall i \in S \tag{3.2}$$

$$\sum_{(i,j) \in E} x_{ij} = b_j \quad \forall j \in D \tag{3.3}$$

$$x_{ij} \in \mathbb{N} \quad \forall (i,j) \in E \tag{3.4}$$

In this chapter, the problem is generalized by associating a colour, either red or blue, to each supply node. Thus, the set of supply nodes is partitioned into two sets R (red) and B (blue) such that $S = R \cup B$, and $R \cap B = \emptyset$. The additional requirement is that the set of supply nodes that actually supply a demand node should all have the same colour. In other words, a demand node is only allowed to receive flow from supply nodes that are either all red or all blue. These constraints are referred to as colour constraints. Obviously, the resulting problem is a generalization of the transportation problem since, if all supply nodes have the same colour, the TP arises. From now on, this problem is denoted as the Red-Blue Transportation Problem (Red-Blue TP).

It is straightforward to relate the Red-Blue TP to a special case of the patient-to-room assignment problem, namely in a static, single day setting. Each patient is represented as a supply node with $a_i = 1$, with gender determining the colour of the supply node; either red or blue. Each room is represented as a demand node where the capacity of the room is represented by b_j, and the "appropriateness" of assigning patient i to room j is captured by cost c_{ij}. Finally, the colour constraints correspond to a gender separation policy.

Although the definition of the Red-Blue TP is motivated by the patient-to-room assignment problem, it is not hard to think of other practical applications. For instance, imagine a situation where a number of goods need to be transported from a port to a warehouse. Several trucks are available for transportation, each driving according to a schedule that fixes the departure times at the port. Depending on delivery deadlines, some truck assignments are more suitable for particular goods than others. A Red-Blue TP instance arises if the goods can be divided into two types that cannot be assigned to the same truck, for instance because of incompatibilities of content (e.g. hazardous materials), ownership (e.g. rivaling business companies that are unwilling to have their goods transported on the same truck), or size [11]. Another application involves transportation of football fans to a match using public railways: when assigning fans to trains, no fans of the opposing teams should be on the same train, to avoid hooliganism [68]. In Section 3.4, further applications of a maximization variant of Red-Blue TP are mentioned.

3.1.1 Related work

The Red-Blue transportation problem is a natural generalization of a classic problem in operations research. In the literature, several generalizations of the transportation problem have been described. The most well-known is probably the transshipment problem, in which the underlying graph does not need to be bipartite and so-called *transferring* nodes, which have no net supply or demand, may exist (see e.g. [59]). The min-cost flow problem is a further generalization of the transshipment problem, introducing capacities on the arcs. In the fixed-charge transportation problem [38], a fixed cost may be incurred for every arc in the transportation network that is used. Numerous other generalizations of the transportation problem have been presented, for instance to solve spatial economic equilibrium problems [50], and aircraft routing problems [27], or even to deal with wartime conditions where distances from some sources to some destinations are no longer definite (i.e. the *grey* transportation problem, see [3]).

One generalized transportation problem is particularly related to the Red-Blue transportation problem, namely the Transportation Problem with Exclusionary Side Constraints (TPESC). Although the name TPESC was coined by Sun [71], it was in fact introduced by Cao [10]. The phenomenon that not every set of supply nodes is allowed to send flow to a demand node, is something that TPESC and Red-Blue TP have in common. In TPESC, for each demand node $j \in D$, a set of pairs of supply nodes is given, denoted by $F_j = \{\{i_1, i_2\}|i_1, i_2 \in S\}$. The problem is to send the flow from supply to demand nodes at minimum cost, such that each demand node $j \in D$ only receives supply from at most one supply node for each pair of supply nodes present in F_j.

It is not hard to see that Red-Blue TP is a special case of TPESC. Goossens and Spieksma [32] show that TPESC is NP-hard, and becomes pseudo-polynomially solvable if the number of supply nodes is fixed. Furthermore, these authors study TPESC with identical exclusionary sets: they provide a pseudo-polynomial algorithm for the case with two demand nodes, and prove NP-hardness for the case with three demand nodes.

Another problem related to Red-Blue TP is the so-called Maximum Flow problem with Conflict Graph (MFCG), a problem studied by Pferschy and Schauer [62]. In the MFCG a directed graph with capacitated arcs, a source, and a sink are given. In addition, pairs of arcs (from the directed graph) are given; for some pairs of arcs the constraint is that at most one arc of the pair can carry flow (a negative disjunctive constraint), for other pairs of arcs the constraint is that at least one arc of the pair must carry flow (a positive disjunctive constraint). Pferschy and Schauer [62] show that the problem of finding a maximum flow in a network under these disjunctive constraints is (strongly) NP-hard; even more they show that no polynomial time constant-factor approximation algorithm can exist (unless P=NP).

Observe that Red-Blue TP is a special case of MFCG; indeed, consider some demand $j \in D$. Now, by having negative disjunctive constraints for each pair of arcs that consist of one arc emanating from a red supply node to node j, and one arc emanating from a blue supply node to node j, an instance of Red-Blue TP arises. Note that for our special case it is possible to find polynomial time constant factor approximation algorithms (see Section 3.4).

3.2 Complexity of Red-Blue TP

As a general statement of the complexity of Red-Blue TP, we provide the following theorem.

Theorem 1. *Red-Blue TP is NP-hard, even if* $a_i = 1$ $\forall i \in S$, *and* $b_j = 3$ $\forall j \in D$.

Proof. We prove Theorem 1 by showing that the EXACT-3-COVER (X3C) problem can be reduced to the decision version of Red-Blue TP. The decision version of Red-Blue TP, denoted Red-Blue TP_D, concerns the question: does there exist a solution that sends all flow from the supply nodes to the demand nodes while satisfying demand, and while satisfying the colour constraints, i.e. does there exist a feasible flow? X3C has been shown to be NP-complete [see e.g. 29], and is defined as follows:

Input: A set X with $|X| = 3q$ and a collection C of 3-element subsets (i.e., triples) of X, with $|C| = k$.

Question: Does there exist a cover in C that covers X exactly, i.e. a sub-collection $C' \subseteq C$ such that every $x_i \in X$ is contained in exactly one $C_j \in C'$?

Any instance of X3C (with $|C| > q$) can be reduced to Red-Blue TP_D as follows. Associate with each element $x_i \in X$ a blue supply node i with $a_i = 1$. Associate with each triple C_j a demand node j with $b_j = 3$. Construct edges from supply to demand nodes corresponding to the membership relations (i.e. supply node x_i is connected to demand node $C_j \Leftrightarrow x_i \in C_j$). Add $3(k-q)$ red supply nodes with $a_i = 1$ that are connected to all demand nodes. Observe that total supply equals total demand. The question is: does there exist a feasible flow in this instance of Red-Blue TP_D?

Next, it is shown that a yes-answer to the X3C instance directly corresponds to a yes-answer to the corresponding Red-Blue TP instance, and vice versa.

First, consider an X3C instance that is feasible, and thus has an exact cover $C' \subseteq C$. Then, each demand node corresponding to a $C_j \in C'$ can be supplied by the blue supply nodes corresponding to the $x_i \in C_j$, and the remaining demand nodes can be supplied by the red supply nodes. Thus, the corresponding Red-Blue TP_D instance is also feasible.

Next, consider any feasible solution to the Red-Blue TP_D instance. Each demand node is supplied by either three red supply nodes or by three blue supply nodes. Moreover, there must exist q demand nodes each supplied by three blue supply nodes. These triples of blue supply nodes correspond to the triples in X3C that form a feasible solution. □

Notice that the above reduction can be generalized to show that Red-Blue TP with $b_j = k$ is at least as hard as Exact Cover by k-sets.

If a cost of zero is put on the edges described in the above proof, and some edges are added with a cost strictly larger than zero (corresponding to $x_i \notin C_j$), a polynomial time algorithm with a constant performance ratio for Red-Blue TP would find a zero cost solution if one exists, and hence would be able to distinguish between the yes-instances and the no-instances of X3C. Therefore, the following corollary holds:

Corollary 1. *There is no polynomial time constant-factor approximation algorithm for Red-Blue TP, even if $a_i = 1$ $\forall i \in S$, and $b_j = 3$ $\forall j \in D$, unless $P = NP$.*

Since the problem setting where, in addition to $a_i = 1$, also $b_j = 1$ reduces to the assignment problem, the only setting for which the complexity is left open is the case where $b_j = 2$.

The special case of Red-Blue TP on a complete bipartite graph also has relevance. Relating back to the patient-to-room assignment problem, patients can also be assigned to unsuitable rooms if necessary, when all other rooms are at capacity. Therefore, the assignment graph is, in this case, complete. The following theorem shows that this does not make the problem easy, even when all edge-costs are equal.

Theorem 2. *Red-Blue TP is NP-hard, even if G is a complete bipartite graph, all edge-costs are equal and there are only 2 supply nodes with equal supply.*

Proof. We prove Theorem 2 by showing that PARTITION can be reduced to Red-Blue TP_D. PARTITION has been shown to be NP-Complete [see e.g. 29] and is defined as follows:

Input: A set of integers $X = \{x_1, x_2, \ldots, x_n\}$ with $\sum_{i=1}^{n} x_i = q$

Question: Does there exist a partition of X into $\{X_1, X_2\}$ such that

$$\sum_{x_i \in X_1} x_i = \sum_{x_j \in X_2} x_j$$

The reduction is as follows. Construct a demand node for each $x_i \in X$ with $b_j = x_i$. Next, construct a blue and a red supply node, each with a supply of $\frac{q}{2}$. Set up edges between each supply/demand node pair so that the resulting bipartite graph is complete. Observe that total supply equals total demand by construction. The question is: does there exist a feasible assignment?

Consider a feasible PARTITION instance with a partition $\{X_1, X_2\}$ of X such that $\sum_{x_i \in X_1} x_i = \sum_{x_j \in X_2} x_j = \frac{q}{2}$. It is clear that the corresponding Red-Blue TP_D instance is also feasible by supplying each demand node $x_i \in X_1$ by the blue node and each demand node $x_j \in X_2$ by the red one.

Next, consider that the Red-Blue TP_D instance is feasible. In any feasible solution, a demand node $x \in X$ will be entirely supplied by either the blue supply node or the red supply node. Since total supply equals total demand, it must be so that in any feasible solution the sum of the demand nodes supplied by blue (red) supply nodes is equal to $\frac{q}{2}$. Therefore, it must be so that when:

$X_1 = \{x \in X |$corresponding demand node is supplied by the *blue* supply node$\}$

$X_2 = \{x \in X |$corresponding demand node is supplied by the *red* supply node$\}$

then:

$$\sum_{x_i \in X_1} x_i = \sum_{x_j \in X_2} x_j = \frac{q}{2}$$

Thus the corresponding PARTITION instance is also feasible.

□

Finally, consider the special case of Red-Blue TP where the number of demand nodes is fixed, but the capacity of the demand nodes is still part of the input. In this case, the following lemma holds:

Lemma 1. *If $|D|$ is fixed, Red-Blue TP is solvable in polynomial time.*

Proof. Red-Blue TP can be seen as a colouring problem, where the demand nodes are to be coloured either blue or red in such a way that all blue (red) supply nodes can be assigned to blue (red) demand nodes. Given a colouring of demand nodes, the feasibility of Red-Blue TP can be determined by solving two transportation problems: the TP on the blue subgraph, and the problem on the red subgraph. If there are $|D|$ demand nodes, then there are $2^{|D|}$ possible colourings of the demand nodes. Thus by solving $2 \times 2^{|D|}$ transportation problems, the feasibility of Red-Blue TP can be determined. Moreover, if a feasible solution exists, this algorithm finds an optimal solution. Since the TP can be solved in polynomial time, this enumeration can be done in time polynomial in the number of supply nodes. □

3.3 Integer models

Two integer programming formulations are presented for Red-Blue TP and it is shown that Formulation 2 is stronger than Formulation 1. Please refer to Table 3.1 for details on the notation.

Notation	Description
S	set of supply nodes,
D	set of demand nodes,
S_j	set of supply nodes that can supply demand node $j \in D$,
D_i	set of demand nodes that can be supplied by node $i \in S$,
a_i	supply of node $i \in S$,
b_j	demand of node $j \in D$,
R (B)	set of red (blue) supply nodes, $S = R \cup B$,
R_j (B_j)	set of red (blue) nodes that can supply demand node $j \in D$,
c_{ij}	the cost of sending one unit of supply from node i to demand node j.

Table 3.1: Notation for describing the Red-Blue TP formulations.

Formulation 1 The decision variables are defined as follows:

$$x_{ij} = \text{amount of supply that node } i \text{ sends to node } j$$

$$y_j = \begin{cases} 1 & \text{if demand node } j \text{ is supplied by red nodes,} \\ 0 & \text{otherwise} \end{cases}$$

Red-Blue TP is modelled as follows:

$$\mathbf{Min} \sum_{i \in S} \sum_{j \in D_i} c_{ij} \cdot x_{ij} \tag{3.5}$$

subject to:

$$\sum_{j \in D_i} x_{ij} = a_i \quad \forall i \in S \tag{3.6}$$

$$\sum_{i \in R_j} x_{ij} = b_j \cdot y_j \quad \forall j \in D \tag{3.7}$$

$$\sum_{i \in B_j} x_{ij} = b_j \cdot (1 - y_j) \quad \forall j \in D \tag{3.8}$$

$$x_{ij} \in \mathbb{N} \quad \forall i \in S, j \in D_i \tag{3.9}$$

$$y_j \in \{0, 1\} \quad \forall j \in D \tag{3.10}$$

The objective function minimizes the total cost of sending supply to demand nodes j. Constraints (3.6) ensure that each supply node i sends its supply a_i to its appropriate demand nodes D_i. Constraints (3.7) and (3.8) ensure that each demand node j receives b_j units of supply from either red or blue supply nodes.

The decision variables x_{ij} are defined for all feasible (i, j) pairs in Expression (3.9). The LP-relaxation of (3.5)-(3.10) arises when we replace (3.9) and (3.10) by $x_{ij} \geqslant 0, \forall i \in S, j \in D_i$, and $0 \leqslant y_j \leqslant 1, \forall j \in D$. The corresponding objective function value is denoted by V_{LP1}.

Formulation 2 The second formulation corresponds to the integer model described by Sun [71]. It uses the same decision variables x_{ij} as Formulation 1, but also uses decision variables y_{ij} that are defined as follows:

$$y_{ij} = \begin{cases} 1 & \text{if node } i \text{ supplies node } j, \\ 0 & \text{otherwise} \end{cases}$$

Red-Blue TP is modelled as follows:

$$\textbf{Min} \sum_{i \in S} \sum_{j \in D_i} c_{ij} \cdot x_{ij} \tag{3.11}$$

subject to:

$$\sum_{j \in D_i} x_{ij} = a_i \quad \forall i \in S \tag{3.12}$$

$$\sum_{i \in S_j} x_{ij} = b_j \quad \forall j \in D \tag{3.13}$$

$$x_{ij} \leq \min(a_i, b_j) \cdot y_{ij} \quad \forall i \in S, j \in D_i \tag{3.14}$$

$$y_{ij} + y_{kj} \leq 1 \quad \forall i \in B, k \in R, j \in D_i \cap D_k \tag{3.15}$$

$$x_{ij} \in \mathbb{N}, y_{ij} \in \{0, 1\} \quad \forall i \in S, j \in D_i \tag{3.16}$$

The objective function is the same as in Formulation 1, minimizing total cost. Constraints (3.12) ensure that each supply node i sends its supply a_i to its appropriate demand nodes D_i, while constraint (3.13) ensures that b_j units of supply are sent to demand node j. Constraints (3.14) express that y_{ij} takes the value 1, when $x_{ij} > 0$. Constraints (3.15) ensure that no red and blue supply node i and k supply the same demand node j.

The LP-relaxation of (3.11)-(3.16) arises when (3.16) is replaced by $x_{ij} \geqslant 0, 0 \leqslant y_{ij} \leqslant 1, \forall i \in S, j \in D_i$. The corresponding objective function value is denoted by V_{LP2}.

Theorem 3. *$V_{LP1} \leqslant V_{LP2}$, namely Formulation 2 is stronger than Formulation 1.*

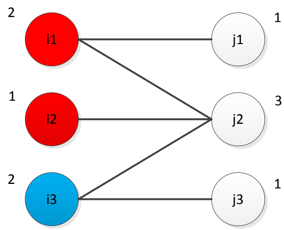


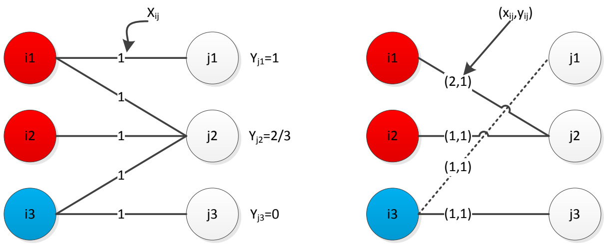
Figure 3.1: A complete bipartite graph with $R = \{i_1, i_2\}$ and $B = \{i_3\}$; $a_{i_1} = 2$, $a_{i_2} = 1$, $a_{i_3} = 2$, $b_{j_1} = 1$, $b_{j_2} = 3$, and $b_{j_3} = 1$. Drawn edges have cost $c_{ij} = 0$, other (not drawn) edges have $c_{ij} = 1$.

Proof. For any instance, take an optimal LP-solution of Formulation 2 $(\mathbf{x_2}, \mathbf{y_2})$ with its value denoted V_{LP2}. A feasible LP-solution of Formulation 1 can be constructed by setting $\mathbf{x_1} = \mathbf{x_2}$ and $y_j = \sum_{i \in R_j} x_{ij}/b_j, \forall j \in D$. It is easy to verify that this is a feasible solution for the LP-relaxation of (3.5)-(3.10) with value V_{LP2}.

Consider the following example showing that Formulation 2 can be better than Formulation 1. Given the complete bipartite graph $G(S \cup D, E)$ with $S = R \cup B$, $R = \{i_1, i_2\}$, $B = \{i_3\}$, $D = \{j_1, j_2, j_3\}$ and $E = S \times D$. Also, $a_{i_1} = a_{i_3} = 2$, $a_{i_2} = 1$, $b_{j_1} = b_{j_3} = 1$, and $b_{j_2} = 3$. All drawn edges in Figure 3.1 have $c_{ij} = 0$, all other edges (not drawn) have $c_{ij} = 1$.

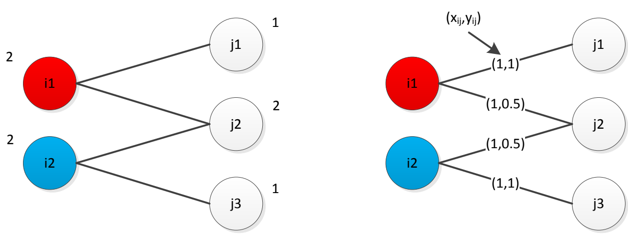
The LP-relaxation of Formulation 1 has an optimal value $V_{LP1} = 0$ (Figure 3.2a), whereas the LP-relaxation of Formulation 2 has an optimal value $V_{LP2} = 1$ (Figure 3.2b). Thus, there are instances for which Formulation 2 is better than Formulation 1. $\square$

Notice that the example in Figure 3.1 also shows that the LP-relaxation of Formulation 1 can be arbitrarily bad compared to the integer optimum. That is also true for Formulation 2: in Figure 3.3a we give an instance with $R = \{i_1\}$, $B = \{i_2\}$ where all drawn edges have cost $c_{ij} = 0$, all other edges (not drawn) have $c_{ij} = 1$. Also, $a_{i_1} = a_{i_2} = 2$, $b_{j_1} = b_{j_3} = 1$, and $b_{j_2} = 2$. This instance has no integer solution with value 0, whereas the LP-relaxation of Formulation 2 allows a solution with value $V_{LP2} = 0$ (Figure 3.3b).



(a) Formulation 1. $V_{LP1} = 0$ (b) Formulation 2. $V_{LP1} = 1$

Figure 3.2: Relaxation of Formulation 1 (left) and Formulation 2 (right). The dashed edge shows that the LP-relaxation of Formulation 2 uses an edge with cost $c_{i_3,j_1} = 1$, resulting in an LP-relaxation value $V_{LP2} = 1$.



(a) Formulation 1. (b) Formulation 2.

Figure 3.3: Example on which Formulation 2 performs arbitrarily bad. A complete bipartite graph, with $R = \{i_1\}$ and $B = \{i_2\}$; $a_{i_1} = a_{i_2} = 2$, $b_{j_1} = b_{j_3} = 1$, and $b_{j_2} = 2$. Drawn edges have cost $c_{ij} = 0$, all other edges (not drawn) have $c_{ij} = 1$. An integer optimum with value 0 does not exist, however, the LP-relaxation of Formulation 2 has an optimal value of $V_{LP2} = 0$ (right).

3.4 The maximization variant of Red-Blue TP

In this section, the following variant of Red-Blue TP is considered: the objective function is modified to maximization, with p_{ij} denoting the profit gained from supplying one unit to demand node j from supply node i. Moreover, we do not insist on sending all flow. This variant of the problem is referred to as Max-Red-Blue TP. In this setting, the patient assignment problem can be seen as assigning as many patients as possible (weighted by e.g. their contribution margin, that may be higher when assigned to a single bed room) to rooms.

An IP-formulation of the maximization variant (corresponding to Formulation 1) is as follows:

$$\mathbf{Max} \sum_{i \in S} \sum_{j \in D_i} p_{ij} \cdot x_{ij} \tag{3.17}$$

subject to:

$$\sum_{j \in D_i} x_{ij} \leq a_i \quad \forall i \in S \tag{3.18}$$

$$\sum_{i \in R_j} x_{ij} \leq b_j \cdot y_j \quad \forall j \in D \tag{3.19}$$

$$\sum_{i \in B_j} x_{ij} \leq b_j \cdot (1 - y_j) \quad \forall j \in D \tag{3.20}$$

$$x_{ij} \in \mathbb{N} \quad \forall i \in S, j \in D_i \tag{3.21}$$

$$y_j \in \{0, 1\} \quad \forall j \in D \tag{3.22}$$

Observe that the LP-relaxation of (3.17)-(3.22) can be found by solving a transportation problem, consisting of (3.17), (3.18), $\sum_{i \in R_j \cup B_j} x_{ij} \leqslant b_j, \forall j \in D$, and $x_{ij} \geqslant 0, \forall i \in S, j \in D_i$. We denote the value of this formulation by V^{max}_{LP1}. This formulation only uses the x-variables; a feasible solution for the y-variables is given by $y_j = \sum_{i \in R_j} x_{ij}/b_j, \forall j \in D$.

It is straightforward to show that Max-Red-Blue TP is NP-hard. Feasibility of Red-Blue TP can be determined by constructing a Max-Red-Blue TP instance with the same bipartite graph and edge profits p_{ij} equal to 1. If the Max-Red-Blue TP admits a solution with flow profit equal to $\sum_{i \in S} a_i$, then the corresponding Red-Blue TP instance is feasible (since all flow can be sent, considering colour constraints).

The Max-Red-Blue TP setting is interesting to study with respect to approximation, as it does not suffer from the feasibility issue. Indeed, in this setting the **0**-vector is always a feasible, be it the worst possible, solution.

In what follows, three approximation algorithms for Max-Red-Blue TP are presented; the final subsection deals with a number of generalizations of Max-Red-Blue TP.

3.4.1 Algorithm 1: MAX-RB

Consider the algorithm described below. It consists of solving two transportation problems and next selecting the best of the two corresponding solutions.

Algorithm 1 MAX-RB

1: Solve a TP on the subgraph induced by $R \cup D$ (the red subgraph), and solve a TP on the subgraph induced by $B \cup D$ (the blue subgraph). The respective solution values are denoted by $V(R)$ and $V(B)$.
2: Return the solution vector for which $\max(V(R), V(B))$ is attained.

Despite its simplicity, the resulting solution vector cannot be arbitrarily bad.

Theorem 4. *The approximation ratio of MAX-RB is $\frac{1}{2}$ and this bound is tight.*

Proof.

$$\max(V(R), V(B)) \geqslant \frac{1}{2}(V(R) + V(B)) \tag{3.23}$$

$$\geqslant \frac{1}{2}OPT. \tag{3.24}$$

with OPT denoting the value of an optimal solution to the Max-Red-Blue TP problem. The first inequality is trivial, the second follows from the observation that $V(R) + V(B)$ is the value of an optimal solution to a relaxed version of the problem, namely one where the demands are doubled (i.e. $b_i := 2b_i$), and where no colour constraints are present.

Finally, it is shown that the bound is tight. Consider the example in Figure 3.4. For this instance, MAX-RB may find $V(R) = V(B) = 2$. However, clearly $OPT = 4$. □

3.4.2 Algorithm 2: TP+R

The algorithm described below (Algorithm 2) consists of solving three transportation problems, one of which corresponds to solving the LP-relaxation

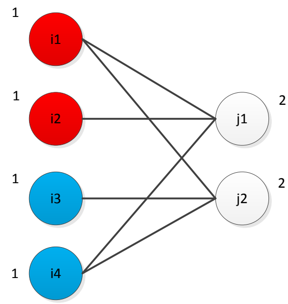


Figure 3.4: A complete bipartite graph with $R = \{i_1, i_2\}$ and $B = \{i_3, i_4\}$; $a_{i_1} = a_{i_2} = a_{i_3} = a_{i_4} = 1$, and $b_{j_1} = b_{j_2} = 2$. Drawn edges have profit $p_{ij} = 1$, other (not drawn) edges have $p_{ij} = 0$.

of (3.17)-(3.22). The algorithm solves the transportation problem on the entire graph. Next it heuristically determines for each demand node j whether it should be supplied by either red or blue supply nodes by evaluating which partial flow (red or blue) is most profitable in the current TP solution. Finally the algorithm solves the TP on the red and blue sub-graphs (induced by $R \cup D_R$ and $B \cup D_B$) to determine the maximum profit flow.

Algorithm 2 TP+R

1: Solve the LP-relaxation of (3.17)-(3.22). Call the resulting solution vector x^*_{ij}. Set $D_R = D_B = \emptyset$.
2: **for all** $j \in D$ **do**
3:     **if** $\sum_{i \in R_j} p_{ij} x^*_{ij} \geqslant \sum_{i \in B_j} p_{ij} x^*_{ij}$ **then**
4:         $D_R := D_R \cup \{j\}$
5:     **else**
6:         $D_B := D_B \cup \{j\}$
7:     **end if**
8: **end for**
9: Solve two TPs, one on the subgraph induced by $R \cup D_R$, and one on $B \cup D_B$ and construct the overall solution with value $V(TP + R)$.

Theorem 5. *The approximation ratio of TP+R is $\frac{1}{2}$ and this bound is tight.*

Proof. Let $V(B \cup D_B)$ and $V(R \cup D_R)$ denote the value of solving the TP on the subgraph induced by $B \cup D_B$ and on the subgraph induced by $R \cup D_R$

respectively. Then:

$$V(TP+R) = V(B \cup D_B) + V(R \cup D_R) \geq \sum_{j \in D_B} \sum_{i \in B_j} p_{ij} x^*_{ij} + \sum_{j \in D_R} \sum_{i \in R_j} p_{ij} x^*_{ij} \tag{3.25}$$

This inequality holds since x^*_{ij} restricted to $j \in D_B$, $i \in B_j$ (or $j \in D_R$, $i \in R_j$) is a feasible solution to the transportation problem solved in line 9 of Algorithm 2. Hence, an optimal solution to that TP has a value at least as large as $\sum_{j \in D_B} \sum_{i \in B_j} p_{ij} x^*_{ij}$.

Define $v_j = \sum_{i \in S_j} p_{ij} x^*_{ij}$ for each $j \in D$. Observe that $V^{max}_{LP1} = \sum_{j \in D} v_j$. By construction of the sets D_R and D_B, it is such that for each $j \in D_R$:

$$\sum_{i \in R_j} p_{ij} x^*_{ij} \geq \frac{1}{2} \left(\sum_{i \in R_j} p_{ij} x^*_{ij} + \sum_{i \in B_j} p_{ij} x^*_{ij} \right) = \frac{1}{2} v_j \tag{3.26}$$

and for each $j \in D_B$:

$$\sum_{i \in B_j} p_{ij} x^*_{ij} \geq \frac{1}{2} \left(\sum_{i \in R_j} p_{ij} x^*_{ij} + \sum_{i \in B_j} p_{ij} x^*_{ij} \right) = \frac{1}{2} v_j \tag{3.27}$$

Thus:

$$\begin{aligned} \sum_{j \in D_B} \sum_{i \in B_j} p_{ij} x^*_{ij} + \sum_{j \in D_R} \sum_{i \in R_j} p_{ij} x^*_{ij} \geq \sum_{j \in D_B} \frac{1}{2} v_j + \sum_{j \in D_R} \frac{1}{2} v_j = \\ \frac{1}{2} \sum_{j \in D} v_j = \frac{1}{2} V^{max}_{LP1} \geq \frac{1}{2} OPT. \end{aligned} \tag{3.28}$$

Finally, it is shown that the bound is tight. Consider again the example in Figure 3.4. In this example, the worst case optimal solution vector to the TP (note that there are several optimal solution vectors) is:

$$x^*_{12} = x^*_{21} = x^*_{32} = x^*_{41} = 1, \quad x^*_{11} = x^*_{22} = x^*_{31} = x^*_{42} = 0. \tag{3.29}$$

Thus:

$$\sum_{i \in R_1} p_{i1} x^*_{i1} = \sum_{i \in B_1} p_{i1} x^*_{i1} = 1, \quad \text{and} \tag{3.30}$$

$$\sum_{i \in R_2} p_{i2} x^*_{i2} = \sum_{i \in B_2} p_{i2} x^*_{i2} = 1. \tag{3.31}$$

Therefore, the worst case colouring is:

$$D_R = \{1, 2\}, \quad D_B = \emptyset. \tag{3.32}$$

Thus, solving the TPs on the subgraphs leads to:

$$V(TP + R) = V(B \cup D_B) + V(R \cup D_R) = 0 + 2 = 2. \tag{3.33}$$

Recall from the previous that $OPT = 4$. □

Corollary 2. *$V_{LP1}^{max} \leq 2 \cdot OPT$.*

Proof. This follows from the above since it is actually shown that $V(TP+R) \geq \frac{1}{2}V_{LP1}^{max}$. Since $OPT \geq V(TP + R)$, the bound follows. □

3.4.3 Algorithm 3: Iterated TP+R

In this variation of TP+R, colours for demand nodes are determined one by one, depending on which colour contributes most to the objective function value in that demand node. Each time a demand node is coloured, the TP is solved again taking this decision into account. This algorithm is denoted as Iterated TP+R (ITP+R).

Algorithm 3 Iterated TP+R

1: Set $D_R = D_B = \emptyset$.
2: Solve a TP based on profits p_{ij}. Call the resulting solution vector x_{ij}^*.
3: Find $j_R = \text{argmax}_{j \in D \setminus (D_R \cup D_B)} \sum_{i \in R_j} p_{ij} x_{ij}^*$ and $j_B = \text{argmax}_{j \in D \setminus (D_R \cup D_B)} \sum_{i \in B_j} p_{ij} x_{ij}^*$.
4: **if** $\sum_{i \in R} p_{ij_R} x_{ij_R}^* \geqslant \sum_{i \in B} p_{ij_B} x_{ij_B}^*$ **then**
5:     set $B_{j_R} := \emptyset$ and $D_R := D_R \cup \{j_R\}$
6: **else**
7:     set $R_{j_B} := \emptyset$ and $D_B := D_B \cup \{j_B\}$
8: **end if**
9: Go to line 2 until $D_R \cup D_B = D$.
10: Solve two TPs, one on the subgraph induced by $R \cup D_R$, and one on $B \cup D_B$, and get the overall solution value $V(ITP + R)$.

Notice that Iterated TP+R consists of solving $|D| + 2$ transportation problems. Although in a practical sense, this results in a good performance (see Section 3.5), from a worst-case point of view, this computational effort does not pay off.

Theorem 6. *The approximation ratio of Iterated TP+R is at most $\frac{1}{2}$.*

Proof. For the example in Figure 3.4, the worst case optimal solution vector to the TP (note that there are several optimal solution vectors) is:

$$x_{12}^* = x_{21}^* = x_{32}^* = x_{41}^* = 1, \quad x_{11}^* = x_{22}^* = x_{31}^* = x_{42}^* = 0. \tag{3.34}$$

Thus, after the first iteration of the algorithm:

$$\sum_{i \in R_1} p_{ij} x_{i1}^* = \sum_{i \in B_1} p_{ij_1} x_{i1}^* = 1, \tag{3.35}$$

$$\sum_{i \in R_2} p_{i2} x_{i2}^* = \sum_{i \in B_2} p_{i2} x_{i2}^* = 1. \tag{3.36}$$

Consider that we set $D_R = \{2\}$. Solving the TP again, the optimal solution vector is:

$$x_{12}^* = x_{21}^* = x_{41}^* = 1, \quad x_{11}^* = x_{22}^* = x_{31}^* = x_{32}^* = x_{42}^* = 0. \tag{3.37}$$

Thus:

$$\sum_{i \in R_1} p_{i1} x_{i1}^* = \sum_{i \in B_1} p_{i1} x_{i1}^* = 1, \tag{3.38}$$

$$\sum_{i \in R_2} p_{i2} x_{i2}^* = 1, \quad \sum_{i \in B_2} p_{i2} x_{i2}^* = 0. \tag{3.39}$$

And finally, $D_R = \{1, 2\}$ and $D_B = \emptyset$. Solving the TPs on the subgraphs results in:

$$V(ITP + R) = V(B \cup D_B) + V(R \cup D_R) = 0 + 2 = 2. \tag{3.40}$$

Again, recall that $OPT = 4$. Thus:

$$\frac{V(ITP + R)}{OPT} = \frac{1}{2} \tag{3.41}$$

□

3.4.4 Generalizations of Max-Red-Blue TP

In this section, a number of generalizations of Max-Red-Blue TP are discussed. One generalization arises when capacities are placed on the edges, i.e. for each edge, the flow transported over it cannot exceed the capacity of the edge. With a small modification (solving *capacitated* transportation problems on the red and the blue subgraph), the MAX-RB algorithm will produce a feasible solution to this problem. Moreover, MAX-RB still has a tight approximation ratio of $\frac{1}{2}$, since the arguments used in Theorem 4 remain valid.

In another generalization, any topology of the underlying graph is allowed. The graph no longer needs to be bipartite; supply nodes, as well as demand nodes, can be joined by an edge. Transferring nodes (i.e. nodes with zero net supply or demand), are allowed as well. The goal is to find a profit-maximizing way to send flow from supply nodes to demand nodes, such that each demand node receives its flow from supply nodes that are either all red or all blue. The MAX-RB algorithm can be adapted as follows: solve an uncapacitated min-cost flow problem on the red (blue) subgraph, i.e. we set the supply of the blue (red) nodes equal to zero, and return the solution vector that results in the highest solution value. Also for this generalization, a tight approximation ratio of $\frac{1}{2}$ holds.

In a third generalization, it is no longer imposed that $R \cap B = \emptyset$, instead the existence of supply nodes that are both red and blue is allowed. These nodes are denoted *colourful*. The colour constraints imply that a demand node can receive supply from either red nodes and colourful nodes, or blue nodes and colourful nodes. Of course, if all supply nodes are colourful, a TP arises. Again, it is easy to see that the MAX-RB keeps its approximation ratio of $\frac{1}{2}$; the transportation problem on the red subgraph, as well as the one on the blue subgraph, now include the colourful supply nodes.

Finally, Max-Red-Blue TP can be generalized to a K-colour variant. In this case, a weighted bipartite graph $G(S \cup D, E)$ is given with $C_1, C_2, \ldots, C_K \subseteq S$, $\bigcup_{k=1}^{K} C_k = S$ and $C_{k_1} \cap C_{k_2} = \emptyset$ for any $1 \leq k_1 < k_2 \leq K$, with $E \subseteq S \times D$. The problem is to find a maximum weighted flow from S to D with respect to supply and demand constraints, and the additional constraint that no two nodes $i_1 \in C_{k_1}, i_2 \in C_{k_2}$, $1 \leq k_1 < k_2 \leq K$ can send flow to the same demand node $j \in D$. The MAX-RB algorithm can be generalized to this setting, by solving transportation problems on the subgraphs induced by $C_k \cup D$, for each colour k. Also, the TP+R algorithm can be generalized to this setting by solving a transportation problem and next identifying which colour gives the largest profit to each demand node. The proofs in Theorem 4 and 5 can trivially be generalized to show that both algorithms guarantee an approximation ratio of at least $\frac{1}{K}$. However, the approximation ratio of the generalization of Iterated TP+R to this K-colour setting remains open.

The following example shows that the bound $\frac{1}{K}$ is tight for both algorithms. Consider a complete bipartite graph $C_1 = \{1, \ldots, K\}$, $C_2 = \{K+1, \ldots, 2K\}$, $\ldots$, $C_K = \{(K-1)K+1, \ldots, K^2\}$ and $D = \{1, \ldots, K\}$. All supply nodes have

$a_i = 1$, all demand nodes have $b_j = K$. Let:

$$\mathbf{1} = \begin{array}{c} 1 \\ 2 \\ \vdots \\ j \\ \vdots \\ K \end{array} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \end{pmatrix}, \quad \mathbf{e_j} = \begin{array}{c} 1 \\ 2 \\ \vdots \\ j \\ \vdots \\ K \end{array} \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix}. \tag{3.42}$$

Then the profit matrix is:

$$(p_{ij}) = \begin{pmatrix} \mathbf{1} & \mathbf{e_{K-1}} & \cdots & \mathbf{e_2} & \mathbf{e_1} \\ \mathbf{e_K} & \mathbf{1} & \cdots & \mathbf{e_2} & \mathbf{e_1} \\ & & \ddots & & \\ \mathbf{e_K} & \mathbf{e_{K-1}} & \cdots & \mathbf{1} & \mathbf{e_1} \\ \mathbf{e_K} & \mathbf{e_{K-1}} & \cdots & \mathbf{e_2} & \mathbf{1} \end{pmatrix} \tag{3.43}$$

Observe that the optimum value equals K^2, which is achieved by sending all supply from the nodes in C_k to demand node k, $k = 1, \ldots, K$. Further, observe that, when applying the generalization of MAX-RB, or the generalization of TP+R, it may happen that the supply of the nodes in each C_k $(1 \leq k \leq K)$ is distributed over the K demand nodes, leading to a value of K.

3.5 Computational study

3.5.1 Experimental setup

We have studied the behaviour of the proposed formulations and approximation algorithms with respect to the following problem characteristics:

- PS: the problem size $(|S| + |D|)$,
- PR: the proportion of red supply nodes $(\frac{|R|}{|S|})$,
- DEN: the density of the graph $(\frac{|E|}{|S| \times |D|})$.

To this purpose, a set of test instances was generated according to a full factorial design with the characteristics described in Table 3.2. The procedure

Parameter	Value
$\text{PS} = \lvert S \rvert + \lvert D \rvert$, with $\lvert S \rvert = \lvert D \rvert$	$\{50, 100, 150, 200, 250, 300, 350, 400\}$
$\text{PR} = \frac{\lvert R \rvert}{\lvert S \rvert}$	$\{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$
$\text{DEN} = \frac{\lvert E \rvert}{\lvert S \rvert \times \lvert D \rvert}$	$\{0.25, 0.5, 0.75, 1.0\}$
S_{max}	50
C_{max}	20

Table 3.2: Characteristics of the generated test instances.

for generating these instances is described in Appendix A.1. The instances are available online (see [76]). Note that for this set of instances $|S| = |D|$. Thus, this experimental setup does not study the effect of the ratio of supply nodes to demand nodes.

For Red-Blue TP, the tightness of the LP-relaxations of Formulations 1 and 2 was tested and compared, as well as the average computation times for each formulation and its linear relaxation. Due to the instance generation procedure, infeasible instances may be generated for which no solution to the integer formulations or the LP-relaxations exists. Thus, the results on these instances are not considered in the forthcoming discussion and table (the number of feasible instances is included in the table).

For the maximization version, Max-Red-Blue TP, the performance of the approximation algorithms, MAX-RB, TP+R and Iterated TP+R, is compared with respect to the integer optimum of Formulation 1. In this setting, feasibility is not an issue as the $\mathbf{0}$-vector is always a feasible solution.

This experimental setup has been coded in the C++ programming language and was compiled with the GNU Compiler Collection (GCC) 4.6.3. The IP-formulations and the LP-relaxation of Formulation 2 were implemented and solved with IBM CPLEX 12.5, using the network simplex algorithm. The LP-formulation of Formulation 1 was implemented as a minimum-cost flow problem, and was implemented and solved with the network simplex algorithm in LEMON 1.3 (**L**ibrary for **E**fficient **M**odeling and **O**ptimization in **N**etworks) from the COIN-OR initiative. The approximation algorithms MAX-RB, TP+R and Iterated TP+R also make use of a minimum-cost flow problem implementation in LEMON 1.3 for solving the transportation problem on the respective (sub)graphs.

All tests were done on a workstation computer equipped with two eight-core Intel Xeon 2670 2.6 GHz processors and 128 GB of main memory (RAM), which was running a Linux-based operating system. The MIP solver was configured to use only one processing thread, so this system was used to solve up to 16 instances in parallel (limiting the MIP solver to 8 GB of memory for each instance).

3.5.2 Results and discussion

Formulation 1 vs Formulation 2

The effect of increasing the problem size (PS) from 50 to 400 nodes is summarized in Table 3.3 for Red-Blue TP. For each setting of PS, the results are averaged over all feasible instances of 240 instances; 10 instances for each of the parameter settings of PR and DEN. It is clear that the computation time of both models increases as the problem size grows, and that the difference in computation time between the two formulations also increases, indicating that Formulation 2 scales worse than Formulation 1. This is expected, as Formulation 2 has a quadratic number of y variables for each source/destination pair, whereas Formulation 1 only has a y variable for each destination node. Furthermore, the exclusionary constraints are also quadratic in number for Formulation 2, whereas they are linear in number for Formulation 1.

For both formulations, the LP-relaxation becomes tighter as the problem size increases; i.e. the gap, defined as $\frac{V_{IP}-V_{LP}}{V_{IP}}$, decreases. Of course, then the difference between the LP-relaxations of both formulations also becomes smaller as the problem size grows. A possible explanation for this observation is that the impact of a single colour constraint decreases when more nodes (or more edges) are present.

The proportion of red vs blue nodes (PR) has a clear impact on the performance of both formulations, shown in Table 3.3. For each setting of PS, the results are averaged over all feasible instances of 320 instances; 10 instances for each of the parameter settings of PS and DEN). It shows that for Red-Blue TP the LP relaxation of Formulation 2 is tighter than Formulation 1, and the difference between the two formulations' tightness increases as the percentage of red nodes grows to 50%. However, as this percentage increases, the computation time for both formulations also increases, and quicker for Formulation 2 than for Formulation 1. At 0% red nodes, both models reduce to the Transportation Problem, and the linear relaxation equals the integer optimum.

Finally, the effect of varying the density of the underlying graph (DEN), from 25% to 100% is also summarized in Table 3.3. For each setting of DEN, the results are averaged over all feasible instances of 480 instances; 10 instances for each of the parameter settings of PS and PR. It is clear that as the density grows to 100%, the gap between the integer optimum and the LP-relaxation of Formulation 1 and 2 decreases. Notice that computation times for Formulation 1 seem to be highest for a 50% density. For Formulation 2, the density for which computation times are highest, is larger, which can be explained by the fact that the number of constraints is directly dependent on the number of edges. Furthermore, the difference between the gap of Formulation 1 and Formulation 2 becomes smaller as the density increases. Thus, the benefit of the stronger LP-relaxation of Formulation 2 again reduces as the density grows to 100%.

It is clear that although Formulation 2 has a tighter LP-relaxation than Formulation 1, this difference decreases as all factors considered (problem size, percentage red nodes, density) grow. Furthermore, in all cases the computation time increases much faster for Formulation 2 than for Formulation 1. This results in a large amount of computation time when trying to find an integer optimum using Formulation 2. Table 3.3 reports the number of instances that timed out after 3600 seconds (these are counted as 3600 seconds in the averages). It shows that for PS $>$ 200 nodes increasingly more instances cannot be solved in less than 1 hour, and in fact (not reported in the table) some instances cannot be solved to optimality in less than 24h. It is also clear that this is highly related to the ratio red/blue nodes. Therefore, we advocate using Formulation 1 over Formulation 2 in all cases.

We can also report the same effects concerning the problem size, proportion of red nodes, and the density for the maximization version of Red-Blue TP. Given that the results show basically the same effects, the corresponding table is not shown.

The approximation algorithms

The computational results for the approximation algorithms are summarized in Table 3.4, showing the relative gap between the heuristic result and the integer optimum, as well as the computation time, while the problem size, the percentage of red nodes, and the density of the network increase.

Overall, the gap with the integer optimum is considerably smaller for the ITP+R heuristic (again, with an exception for the special case where all supply nodes have the same colour) than both MAX-RB and TP+R. MAX-RB, obviously being the most naive algorithm, performs worst of all and almost reaches its worst case behaviour on instances where the percentage of red nodes is equal to 50%. Note that none of these heuristics dominates any of the other heuristics. For each of the heuristics, instances were found for which it outperforms the others.

Param.	Value	# Feasible instances	IP Formulation 1	LP Formulation 1		IP Formulation 2	LP Formulation 2	
			T (s)	Gap (%) $\frac{V_{IP}-V_{LP1}}{V_{IP}}$	T (s)	T (s) (**# timeout after 3600s**)	Gap (%) $\frac{V_{IP}-V_{LP2}}{V_{IP}}$	T (s)
PS (#)	50	187	0.1	9.5	< 0.1	0.7 (**0**)	4.2	< 0.1
	100	236	0.4	8.6	< 0.1	5.7 (**0**)	3.4	0.1
	150	240	1.7	7.4	< 0.1	47.9 (**0**)	2.8	0.2
	200	240	5.4	6.6	< 0.1	300.5 (**0**)	2.4	0.9
	250	240	12.0	5.8	< 0.1	949.7 (**18**)	2.1	2.7
	300	240	19.3	4.7	< 0.1	1393.3 (**42**)	1.6	6.3
	350	240	30.7	4.0	< 0.1	1817.6 (**70**)	1.3	12.2
	400	240	48.6	3.5	< 0.1	1993.7 (**82**)	1.1	22.0
PR (%)	0	315	0.2	0.0	< 0.1	0.2 (**0**)	0.0	0.1
	10	309	6.0	4.6	< 0.1	392.8 (**5**)	2.3	4.1
	20	307	12.3	6.6	< 0.1	918.5 (**27**)	2.6	6.0
	30	312	17.8	7.9	< 0.1	1132.3 (**51**)	2.9	7.2
	40	310	25.8	8.8	< 0.1	1273.8 (**60**)	3.0	8.4
	50	310	29.4	9.0	< 0.1	1324.3 (**69**)	3.0	8.7
DEN (%)	25	425	15.9	8.4	< 0.1	655.5 (**42**)	3.2	0.6
	50	478	24.3	7.2	< 0.1	945.8 (**81**)	2.7	3.1
	75	480	14.1	5.2	< 0.1	1007.2 (**69**)	1.9	7.0
	100	480	6.7	4.1	< 0.1	725.1 (**20**)	1.5	11.7

Table 3.3: Comparison of the gap from the integer optimum, and the computation time for Formulation 1 and Formulation 2. Results shown w.r.t. the influence of the size (PS) of the graph, the proportion of red nodes (PR) and the density (DEN) of the graph, averaged over 10 runs of all values of the respective other parameters.

Clearly, for ITP+R the number of transportation problems that need to be solved increases with the number of demand nodes. Therefore, its computation time increases faster than with TP+R, where only three transportation problems need to be solved, irrespective of the problem size.

3.6 Conclusion

The Red-Blue TP is a very natural generalization of the transportation problem, namely where supply nodes receive one of two colours, and demand nodes cannot receive flow from supply nodes with different colours. The problem definition was introduced to study the computational complexity of patient-to-room assignments under a gender separation policy. The complexity status of the problem was shown to be NP-hard, and it is shown that a constant-factor approximation is not likely to exist, even in a number of special cases. Relating back to the patient-to-room assignment problem, it is clear that the gender separation constraint by itself is sufficient to make the problem difficult (from a computational complexity point of view).

Two IP formulations for the problem were presented: although it is shown that one formulation is strictly stronger than the other, experimental results show that the stronger formulation requires increasingly more computation time than the weaker formulation, as the problem size, the percentage of red nodes, or the density of the graph increase.

Furthermore, a maximization variant of Red-Blue TP was introduced and its approximability was studied. Three algorithms were developed, two of which guarantee an approximation ratio of $\frac{1}{2}$. Computational experiments show that Iterated TP+R achieves the best approximation on most instances, at the expense of considerable computation times. Finally, a number of generalizations of Max-Red-Blue TP were discussed, including a variant with K colours.

		MAX-RB		TP+R		ITP+R		IP (Formulation 1)
Param.	Value	Gap (%)	T (s)	Gap (%)	T (s)	Gap (%)	T (s)	T (s)
PS (#)	50	19.9	< 0.1	2.4	< 0.1	0.4	< 0.1	0.1
	100	22.2	< 0.1	2.7	< 0.1	0.2	0.1	0.6
	150	22.9	< 0.1	2.6	< 0.1	0.2	0.2	2.2
	200	23.8	< 0.1	2.6	< 0.1	0.1	0.5	5.1
	250	23.8	< 0.1	2.3	< 0.1	0.1	1.0	10.0
	300	23.9	< 0.1	2.3	< 0.1	0.1	1.6	16.8
	350	24.1	< 0.1	2.1	< 0.1	0.1	2.6	28.9
	400	24.5	< 0.1	2.3	< 0.1	0.1	3.9	53.3
PR (%)	0	0.0	< 0.1	0.0	< 0.1	0.0	< 0.1	0.1
	10	8.4	< 0.1	2.2	< 0.1	0.2	1.6	5.9
	20	18.3	< 0.1	3.2	< 0.1	0.2	1.5	12.1
	30	28.1	< 0.1	3.3	< 0.1	0.2	1.5	18.4
	40	38.1	< 0.1	3.0	< 0.1	0.2	1.4	27.1
	50	46.0	< 0.1	2.9	< 0.1	0.2	1.4	24.2
DEN (%)	25	21.7	< 0.1	2.0	< 0.1	0.3	0.8	17.2
	50	23.0	< 0.1	2.4	< 0.1	0.2	1.2	23.3
	75	23.6	< 0.1	2.6	< 0.1	0.1	1.4	11.9
	100	24.2	< 0.1	2.6	< 0.1	0.1	1.6	6.0

Table 3.4: Comparison on the gap from the integer optimum and the computation time for the TP+R, ITP+R and MAX-RB heuristics, averaged over 10 runs of all values of the respective other parameters. Results shown w.r.t. the influence of the problem size (PS), the percentage of red nodes (PR) and the density (DEN) of the graph.

Chapter 4

Robust surgery admission scheduling in an online setting

The PA problem that has been the focus of the previous two chapters is essentially concerned with a spatial decision: the problem is to determine *where* patients will stay during their hospitalization. As discussed, this decision clearly has an impact on the quality of care as well as patients' comfort. However, the overall flow of patients and the 'load' across the hospital remains unchanged, as the admission dates are considered an input. Undoubtedly, the patient load has an important impact on hospital operations. Therefore, this chapter focuses on the decision on *when* patients should be admitted.

A robust admission scheduling approach for determining admission dates of elective surgical patients is presented. The admission scheduling problem is modelled in an online, stochastic decision making setting. The aim is to minimize expected operating theatre costs and waiting time, while avoiding the risk of bed shortages at a fixed certainty level by means of a chance-constrained formulation. This stochastic model is approximated by means of sample average approximation and is solved by a meta-heuristic algorithm. The approach is used to implement four admission scheduling strategies that are evaluated based on different criteria in a computational study.

4.1 Introduction

Faced with the uncertain and highly variable nature of the care process, hospital managers try to control the flow of patients to their best extent. Their aim is to provide care to all patients in a timely fashion and to maximize the efficient usage of their most costly resources such as the operating theatre (OT). However, operational inefficiencies such as overtime in the operating theatre, bed shortages in hospital wards, increase costs and are to be avoided.

One key instrument in this management process is admission scheduling of elective patients. Contrary to urgent or emergency patients, by its nature the flow of elective patients is the only flow that can be planned and thus can be ‘tuned’ to the available capacity of the operating theatre, nursing wards and other resources. However, this requires careful planning and an overview of availability of all considered resources, forecasts on urgent/emergency admissions and other uncertain factors.

In practice, the admission scheduling for individual patients is mostly performed by manual planners (e.g. an admission office). The request for admission typically follows from a physician or surgeon who has determined during a consultation that a patient needs to be admitted for treatment. One observation that can be made is that the manual admission scheduling process is often myopic. For example, a surgeon may schedule surgery on an elective patient by considering her/his available operating time in the OT to determine a surgery date. However, assuming that the patient may need to recover a few days, the patient will need to be admitted to a bed for that duration. The availability of a free bed is often assumed rather than really considered. Therefore, a bed shortage may occur when occupancy is high, requiring an ad-hoc operational intervention (discharging a patient earlier to free a bed, admitting the patient to wards of a different discipline, postponing/cancelling the patient’s admission). Another complicating factor is the uncertainty on how long patients will take to recover from treatment. Although a physician or surgeon may have an estimate, every patient is different and complications may cause further variability. Considering future bed availability may therefore be difficult and is easily over- or underestimated.

This chapters presents an admission scheduling approach that employs stochastic information and sampling to consider variations in patients’ recovery times and surgery durations. The risk of bed shortages is avoided at a fixed certainty level. The focus is on the admission scheduling of elective surgical patients, as described in the previous example. The motivation for this consideration follows from the fact that one of the largest patient admission flows is that of patients being admitted for surgery. As already noted in the introduction (Chapter 1), as high as 60% of hospital admissions may be surgery related [74]. The surgical patient flow is of particular interest as it involves the hospital's operating theatre. It can be considered as the prime hospital resource, generating significant revenue, though, due to its resource intensive nature (personnel, equipment), at considerable cost. Another large patient flow is that of patients being admitted for examinations or medical treatment; this flow is not considered in this chapter.

Using this *robust* surgery admission scheduling approach, four different admission scheduling strategies are developed and compared in an online decision making setting. These four strategies differ in both the scheduling horizon they assume, as well as the frequency of decision making. Consequently, these four strategies have different levels of flexibility for optimization and will therefore achieve different levels of performance with respect to hospital ward bed shortages, operating theatre usage cost and patient waiting time.

4.1.1 Related work

As noted by Hulshof et al. [40], considerable attention has been devoted to strategic and tactical decisions for inpatient care services. Particularly, capacity dimensioning and allocation of wards/beds and staff have been well studied with a variety of techniques (simulation, mathematical programming, markov processes, queueing theory).

Concerning admission scheduling and surgical ward usage, studies have been performed mainly in the context of operating theatre planning and scheduling since, as already noted, surgery is a major driving factor for admission. Notably the connection between surgical scheduling and the impact on bed usage in surgical wards has been considered at the tactical decision level. A prime example is the development of a master surgical schedule (MSS). The MSS is a (typically cyclical) timetable that allocates operating rooms and time slots to individual surgeons/surgical disciplines based on their allocated (bi-)weekly capacity. The distribution of these OT slots in the MSS can be related to the bed usage in the surgical wards (through consideration of the length of stay (LOS) distributions and arrival patterns of the different surgical disciplines, see e.g. [5, 28, 73, 75]). By manipulation of the MSS, it is possible to reduce variation in the bed usage and possibly reduce the number of required beds.

Another approach to admission scheduling at the tactical decision level concerns the development of admission control schemes to set admission quota. These admission quota serve as rules by which admission planners can assign individual admissions. For example, Bekker and Koeleman [4] investigated approximations for determining the impact of daily variability in admissions on the variability in bed demand and blocking probabilities. Using these approximations, Bekker and

Koeleman employed a quadratic programming model to determine an admission scheme that minimizes weighted deviations of the bed usage from a defined target load. Hulshof et al. [41] developed a mixed integer linear programming (MILP) model to construct tactical resource allocations and admission plans. The model is able to determine a selection of patients to be served that are in a particular stage of their care process. Their main aim was to achieve equitable access for patients, to meet admission targets and to use resources efficiently.

At the operational decision level, admission scheduling is concerned with the determination of admission dates for individual patients. Vissers et al. [81] developed a platform for comparing different admission strategies, considering resources such as the operating theatre, nursing requirements, bed usage and ICU usage. Different admission strategies such as *maximum resource usage* (MRU) that employ waiting lists, *booked admissions* (determining an admission date at request time) and *zero wait* (admitting a patient at request time) were compared with respect to several performance measures. Mazier et al. [55] discussed the problem of scheduling inpatient admissions in a hospital with a highly uncertain length of stay and a significant number of emergency admissions. Their main concern was to assure enough beds are available for unknown emergency patients and future unknown elective patients. No other resources are considered. To this end, Mazier et al. modelled the problem as a stochastic programming problem and proposed and studied different estimation techniques for assessing the number of beds required by emergency patients. Schmidt et al. [67] presented an admission scheduling decision support system that determines both admission dates as bed assignments for elective patients. The system considers patient priorities, gender and preferences, the availability of LOS estimates and dynamic adjustment of the LOS estimate if patients stay longer than expected. Both an exact method and heuristic methods are presented for the decision support system and are compared through simulation. Gartner and Kolisch [30] presented an approach to schedule the hospital wide patient flow at the operational level. Patient admissions are assumed to be classified by their diagnosis-related group (DRG) and their clinical pathway. The clinical pathway gives a blueprint of the sequence of activities, such as diagnostic activities, surgery, that will occur during a patient's stay, as well as their corresponding resource requirements. Two mixed-integer programming models are presented that aim to maximize the total contribution margin of performing these activities, one in which admission dates are assumed to be fixed/given and one in which admission dates are decision variables. The models are tested and compared on real world data from a mid-size hospital in both a static as a rolling horizon approach.

For surgical patients, the admission date is mostly determined by their assigned surgery date. Considerable attention has been devoted to surgery scheduling at the operational level (see e.g. Cardoen et al. [13] for a thorough review). As operating theatre planning and scheduling is the subject of Chapter 5, most related work is discussed there. However, one observation that can be made is that very few of these works consider the bed usage caused by surgical scheduling at surgical wards. One notable exception is the consideration of the intensive care unit (ICU), which is considered a bottleneck resource for certain surgical types. Min and Yih [57] presented a model to determine an optimal surgery schedule for elective patients, considering uncertainty on surgery durations and LOS in downstream care units (specifically the ICU). A stochastic optimization approach using the sample average approximation method (SAA) is proposed and it is shown to be superior at minimizing overtime to using solely estimated values (the expected value). However, this comes at the cost of additional underutilization.

4.1.2 Contribution

This chapter presents a stochastic optimization model for elective surgery admission scheduling. The model extends Min and Yih [57]'s approach to additionally apply chance-constrained bed usage constraints to minimize risk on bed shortages. Whereas the model of Min and Yih avoid ICU bed shortages altogether, the current model is parametrizable with respect to bed shortage risk. In addition, as the LOS in surgical wards is typically longer than the ICU LOS, the model is also extended to use a longer scheduling horizon to plan all patients, rather than assuming a lower bound cost for unplanned patients. Because of these two complicating factors, a heuristic approach based on local search is developed to solve this chance-constrained surgical admission scheduling problem. This approach is then used to develop different admission scheduling strategies that resemble the concepts explored by Vissers et al. [81]. These strategies are compared in a computational study on a dataset that was generated according to fitted distributions from a hospital.

4.2 Problem definition

4.2.1 The offline problem

This chapter considers a setting in which a set of elective surgical patients P of a certain surgical discipline must be admitted to a hospital for treatment over a certain scheduling horizon H'. Each patient $p \in P$ requires some surgical procedure to be performed and requires some days for recovery in a surgical ward before being discharged. The surgical discipline has a certain ward capacity, denoted $BEDS$, that can be used for admitting patients. In addition, the surgical discipline can make use of a set of operating rooms O for performing surgeries, in which their daily allocated surgery capacity is denoted by CAP_{ot} (possibly determined by an MSS).

Each patient p is characterised by:

- the time at which the request for admission was made r_p,
- the earliest possible date for admission rd_p,
- the number of days before surgery the patient must be admitted $preop_p$,
- the duration of the surgical procedure d_p,
- the length of the recovery period los_p,

The aim is to provide to each patient an admission date a_p such that he/she can be admitted to a bed (i.e. a bed must be available) and two performance measures are optimized:

1. Operating theatre cost: irrespective of individual surgery/treatment costs and profits (i.e. some treatments may be more profitable than others), the aim is to utilize the available operating theatre capacity as good as possible. Deviation from this available capacity should be minimized as both underutilization and running into overtime increase operating costs: underutilization will decrease revenue and incur opportunity costs, whereas running into overtime will incur additional staffing costs and mostly at a higher rate.
2. Patient waiting time: a secondary aim is to admit patients in a timely fashion. Although elective patients are considered not urgent, long waiting times may increase the risk of a deteriorating medical condition and thus incur additional costs. Moreover, patient satisfaction is reduced as patients have to wait longer.

The previous problem definition may be modelled by a mixed integer linear programming formulation as follows. Define the following decision variables:

$$x_{pt} = \begin{cases} 1 & \text{if patient } p \text{ is admitted at time } t, \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

$$y_{pto} = \begin{cases} 1 & \text{if patient } p \text{ undergoes surgery at time } t \text{ in OR } o, \\ 0 & \text{otherwise.} \end{cases} \tag{4.2}$$

The objective of the model is to minimize and balance operating theatre costs, and patient waiting time. Let ot^o_{ot},ot^u_{ot} denote real variables representing the overtime resp. undertime in operating room o at time t. The objective is then:

$$\text{Minimize } W_{\text{OT}} \cdot \sum_{1\leq t\leq H'} \sum_{o\in O} \alpha \cdot ot^o_{ot} + ot^u_{ot}$$
$$+W_{\text{WAIT}} \cdot \sum_{p\in P} \sum_{rd_p\leq t\leq H'} (t - rd_p) \cdot x_{pt} \quad (4.3)$$

with W_{OT}, W_{WAIT} denoting weights reflecting the relative importance of the different costs and α denoting the cost ratio of overtime with respect to underutilization.

The model is subject to the following constraints. Firstly, the following constraints enforce that each patient is admitted exactly once and operated once:

$$\sum_{rd_p\leq t\leq H'} x_{pt} = 1 \qquad \forall p \in P \quad (4.4)$$

$$\sum_{o\in O} y_{p\tau o} = x_{pt} \qquad \forall p \in P, rd_p \leq t \leq H', \tau = t + preop_p \quad (4.5)$$

Secondly, the following relates for each operating room o and each day t of the scheduling horizon, the surgical load to the deviations (overtime/undertime) from the operating room's target capacity CAP_{ot}.

$$\sum_{p\in P} d_p \cdot y_{pto} - CAP_{ot} \leq ot^o_{ot} \qquad \forall o \in O, 1 \leq t \leq H' \quad (4.6)$$

$$CAP_{ot} - \sum_{p\in P} d_p \cdot y_{pto} \leq ot^u_{ot} \qquad \forall o \in O, 1 \leq t \leq H' \quad (4.7)$$

In an ideal setting, the admission scheduling should be such that all elective patients that are admitted for surgery can be admitted to the hospital ward. Therefore, the following expression relates the admission scheduling to its corresponding bed usage and constrains it to be at most equal to the available bed capacity:

$$\sum_{p\in P} \sum_{rd_p\leq \tau\leq t<\tau+los_p} x_{p\tau} \leq BEDS \qquad \forall 1 \leq t \leq H' \quad (4.8)$$

Third, for each patient, decision variables are constrained to the relevant days of the scheduling horizon, i.e. after each patient's earliest admission date rd_p:

$$x_{pt} = 0 \qquad \forall p \in P, 1 \leq t < rd_p \quad (4.9)$$

$$y_{pto} = 0 \qquad \forall p \in P, o \in O, 1 \leq t < rd_p + preop_p \quad (4.10)$$

Finally, the domains of the decision variables x_{pt}, y_{pt} and aid variables ot^o_{ot}, ot^u_{ot} are defined.

$$x_{pt} \in \{0,1\} \qquad \forall p \in P, 1 \leq t \leq H' \quad (4.11)$$

$$y_{pto} \in \{0,1\} \qquad \forall p \in P, o \in O, 1 \leq t \leq H' \quad (4.12)$$

$$ot^o_{ot}, ot^u_{ot} \geq 0 \qquad \forall o \in O, 1 \leq t \leq H' \quad (4.13)$$

4.2.2 The online problem

Clearly, the previous problem formulation and mixed integer programming model is not applicable to *online* decision making as it assumes knowledge of uncertain parameters. The release date rd_p, the surgery duration d_p and the length of stay los_p of all patients $p \in P$ only become known as patient admission requests arrive, patients undergo surgery and finally leave the hospital. Therefore, advance knowledge of these uncertain parameters cannot be assumed *a priori*. Decision making can only be done in an online setting, making decisions as new events occur. However, this does not imply that decision making must be done instantaneously. Decisions on when to admit patients may be postponed (e.g. put on a waiting list) and gathered to make a globally better decision than in an instantaneous decision making setting. Another implication that should not be made is that there is no knowledge at all of uncertain parameters. Length of stay of a patient may not be known until a patient is discharged, but certainly information is available on previous patients' length of stay that have undergone a similar treatment. A similar argument holds for the surgery duration. Therefore, knowledge on the expected length of stay and surgery duration of any patient may be assumed, as well as some idea of their distribution/spread.

Consider a decision time t' at which patient admissions are to be scheduled for the upcoming scheduling horizon H. At this point, a set of patients has already been given an admission date (and are possibly already admitted to the hospital ward), whilst a waiting list of patient requests for admission has accumulated. Let $A_{t'} \subseteq P$ denote the patients that have already received an admission date, i.e. $A_{t'} = \{p \in P | r_p \leq t' \text{ and } a_p \neq \emptyset\}$ and let $P_{t'} \subseteq P$ denote patients that are in the waiting list, i.e. $P_{t'} = \{p \in P | r_p \leq t' \text{ and } a_p = \emptyset\}$.

For these patients $p \in P_{t'}$, assume the following is known:

- the earliest possible date for admission rd_p,

- the number of days before surgery the patient must be admitted $preop_p$,
- the distribution of the duration of p's surgical procedure,
- the distribution of p's length of stay.

In addition, at decision time t' information on departures of admitted patients becomes known if departures occur (i.e. the true LOS of any admitted patient is only known at time of departure).

The aim for the online problem is now to find a *policy* or an admission scheduling strategy that minimizes expression (4.3) over the entire scheduling horizon. An additional difficulty is that bed shortages cannot be avoided. Due to variability in length of stay and arrivals, considerable variation in bed usage may occur. As it is (financially) impossible to dimension wards such that no bed shortages can occur, bed shortages should be avoided as much as possible.

4.3 Robust surgery admission scheduling

A heuristic approach to the *online* surgery admission scheduling problem under uncertainty of both surgery durations and length of stay is presented. To be robust against uncertainty in patient's LOS and to control the risk of bed shortages, a chance-constrained optimization model is presented. A sample average approximation of this model is discussed and addressed by local search. Finally, four admission scheduling strategies are presented that employ the chance-constrained model in their decision making.

4.3.1 Chance-constrained surgery admission scheduling

As already discussed in the previous section, in an online decision making setting, we can only assume the availability of stochastic information (i.e. the distribution) on the surgery duration and length of stay of the individual patients that need to be planned. However, this poses a problem if the aim is to ensure that each patient can be admitted to a surgical ward bed. Although it is possible to plan admissions considering a worst possible LOS estimate ζ_p^{MAX}, this would likely result in an extreme underutilization of the surgical ward. LOS distributions generally have a positively skewed shape with typically a long tail. Scheduling with such worst case LOS estimates would result in a bed usage pattern that is extremely unlikely. Another possibility is to schedule patient admissions considering their expected LOS. However, this may over/underestimate the bed usage as it has no notion of the variance of the LOS.

To control the risk of bed shortages, a chance-constrained approach is more appropriate. Chance-constrained optimization was introduced by Charnes et al. [17] in the context of scheduling heating oil production under uncertainty of demand. Probabilistic constraints are imposed to ensure that sales requirements and inventory storage constraints are met with a certain confidence.
For admission scheduling, such an approach can take into account the distribution of the daily bed usage (that is a function of the admission scheduling and LOS distributions of the patients) and limit that bed usage to the available capacity at a predetermined risk/certainty level.

Consider again model (4.3)-(4.13). It can be adapted to a chance-constrained optimization model in a straightforward manner. Let $\xi = (\xi_{p_1}, \xi_{p_2}, \ldots, \xi_{p_N})$ denote a vector of random variables representing the surgery durations of the patients considered and similarly let $\zeta = (\zeta_{p_1}, \zeta_{p_2}, \ldots, \zeta_{p_N})$ denote a random vector representing the patients' LOS.

The decision variables of the chance-constrained model remain the same, i.e.:

$$x_{pt} = \begin{cases} 1 & \text{if patient } p \text{ is admitted at time } t, \\ 0 & \text{otherwise.} \end{cases} \tag{4.14}$$

$$y_{pto} = \begin{cases} 1 & \text{if patient } p \text{ undergoes surgery at time } t \text{ in OR } o, \\ 0 & \text{otherwise.} \end{cases} \tag{4.15}$$

To consider the uncertain nature of the operating theatre usage, the objective of the model is changed to minimize and balance the *expected* operating theatre costs and patient waiting time. By doing so, the aim is to minimize the operating theatre cost in the average case. Assume that we can consider all possible realizations of random vector ξ, represented by $\xi^k \in \Xi$ with $\Xi \subseteq \mathbb{R}^N$ denoting the support of the probability distribution of ξ.

Let $ot^o_{ot}(\xi^k)$, $ot^u_{ot}(\xi^k)$ denote positive real-valued variables representing the overtime resp. undertime in operating room o at time t, given a particular realization ξ^k. The objective is then:

$$\begin{aligned} \text{Minimize } W_{\text{OT}} \cdot \mathbf{E}_{\Xi} \left[\sum_{t' \leq t \leq H'} \sum_{o \in O} \alpha \cdot ot^o_{ot}(\xi^k) + ot^u_{ot}(\xi^k) \right] \\ + W_{\text{WAIT}} \cdot \sum_{p \in P_{t'}} \sum_{rd_p \leq t \leq H'} (t - rd_p) \cdot x_{pt} \end{aligned} \tag{4.16}$$

with $\mathbf{E}_{\Xi}[\ldots]$ denoting the expected value of $(\ldots)$ over all possible realizations of ξ. Note that in the online setting only patients in the waiting list $P_{t'}$ are scheduled.

The model is subject to the following constraints. Firstly, the following constraints enforce that each patient is admitted exactly once and operated once:

$$\sum_{rd_p \le t \le H'} x_{pt} = 1 \qquad \forall p \in P_{t'} \tag{4.17}$$

$$\sum_{o \in O} y_{p\tau o} = x_{pt} \qquad \forall p \in P_{t'}, rd_p \le t \le H', \tau = t + preop_p \tag{4.18}$$

Secondly, the following relates for each operating room o and each day t of the scheduling horizon, the surgical load to the deviations (overtime/under-utilization) from the operating room's target capacity CAP_{ot}. In the online setting, the expressions considers both patients that are still to be scheduled and patients in $A_{t'}$ that are already admitted (if $a_p \le t'$) or whose future admission date is already fixed (if $a_p > t'$).

$$\sum_{p \in P_{t'}} \xi_p^k \cdot y_{pto} + \sum_{\substack{p \in A_{t'} \\ t = a_p + preop_p}} \xi_p^k - CAP_{ot} \le ot_{ot}^o(\xi^k) \qquad \forall o \in O, t' \le t \le H', \quad \xi^k \in \Xi \tag{4.19}$$

$$CAP_{ot} - \sum_{\substack{p \in A_{t'} \\ t = a_p + preop_p}} \xi_p^k - \sum_{p \in P_{t'}} \xi_p^k \cdot y_{pto} \le ot_{ot}^u(\xi^k) \qquad \forall o \in O, t' \le t \le H', \quad \xi^k \in \Xi \tag{4.20}$$

To limit hospital ward usage for each day of the scheduling horizon, a chance-constraint is introduced to constrain usage to be below or equal to the available number of beds with probability η:

$$\mathbf{Pr}\left\{ \sum_{\substack{p \in A_{t'} \\ a_p \le t < a_p + \zeta_p}} 1 + \sum_{p \in P_{t'}} \sum_{rd_p \le \tau \le t < \tau + \zeta_p} x_{p\tau} \le BEDS \right\} \ge \eta \qquad \forall t' \le t \le H' \tag{4.21}$$

where $\mathbf{Pr}\{\ldots\}$ denotes the probability of event $(\ldots)$ occurring.

Again, for each patient, decision variables are constrained to the relevant days of the scheduling horizon, i.e. after each patient's earliest admission date rd_p:

$$x_{pt} = 0 \qquad \forall p \in P_{t'}, t' \leq t < rd_p \tag{4.22}$$

$$y_{pto} = 0 \qquad \forall p \in P_{t'}, o \in O, t' \leq t < rd_p + preop_p \tag{4.23}$$

Finally, the domains of decision variables x_{pt}, y_{pto} and aid variables $ot^o_{ot}(\xi^k)$, $ot^u_{ot}(\xi^k)$ are defined:

$$x_{pt} \in \{0,1\} \qquad \forall p \in P_{t'}, t' \leq t \leq H' \tag{4.24}$$

$$y_{pto} \in \{0,1\} \qquad \forall p \in P_{t'}, o \in O, t' \leq t \leq H' \tag{4.25}$$

$$ot^o_{ot}(\xi^k), ot^u_{ot}(\xi^k) \geq 0 \qquad \forall o \in O, t' \leq t \leq H', \xi^k \in \Xi \tag{4.26}$$

4.3.2 Sample average approximation

Formulation (4.16)-(4.26) is difficult to solve in general as it depends on random vectors ξ and ζ and all their possible realizations (which may be infinite). Therefore, sample average approximation (SAA) [45] can be used to approximate the stochastic model, transforming it in to a deterministic one. Informally, SAA approximates a stochastic programming problem such as model (4.16)-(4.26), by considering a sample of realizations of ξ and ζ of size K to represent the entire set of possible outcomes. Each realization (ξ^k, ζ^k) is taken with a probability $p_k = \frac{1}{K}$. The expected value of a function $F(x, \xi, \zeta)$ can then be approximated by:

$$\mathbf{E}\left[F(x, \xi, \zeta)\right] \sim \frac{1}{K} \sum_{k=1}^{K} F(x, \xi^k, \zeta^k) \tag{4.27}$$

As $K \to \infty$, the SAA will converge to the true expected value. The application of SAA for chance-constraints is more recent [61]. The main idea is to rewrite a probabilistic constraint:

$$\mathbf{Pr}\left\{G(x, \xi, \zeta) \geq 0\right\} \geq \eta \tag{4.28}$$

to the equivalent form:

$$\mathbf{E}\left[\mathbb{1}_{(0,\infty)}(G(x, \xi, \zeta))\right] \geq \eta \tag{4.29}$$

with:

$$\mathbb{1}_{(0,\infty)}(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases} \tag{4.30}$$

an indicator function of the positive real numbers. Expression (4.29) can then be approximated by:

$$\mathbf{E}\left[\mathbb{1}_{(0,\infty)}(G(x,\xi,\zeta))\right] \geq \eta \sim \frac{1}{K}\sum_{k=1}^{K}\mathbb{1}_{(0,\infty)}(G(x,\xi^k,\zeta^k)) \geq \eta \tag{4.31}$$

For a more thorough introduction, refer to e.g. [69].

Let $\xi^1, \xi^2, \ldots, \xi^K$ be K i.i.d. samples of the random vector ξ and similarly $\zeta^1, \zeta^2, \ldots, \zeta^K$ be K i.i.d. samples of random vector ζ. Introduce the following decision aid variables to serve as indicator variables for exceeding bed capacity:

$$z_{tk} = \begin{cases} 1 & \text{if bed usage in sample } k \text{ exceeds the available capacity} \\ & \text{at time } t, \\ 0 & \text{otherwise.} \end{cases} \tag{4.32}$$

The SAA of model (4.16)-(4.26) is then defined by:

$$\text{Min } W_{\text{OT}} \cdot \sum_{k=1}^{K} \frac{1}{K} \sum_{t' \leq t \leq H'} \sum_{o \in O} \alpha \cdot ot_{ot}^{o}(\xi^k) + ot_{ot}^{u}(\xi^k)$$

$$+W_{\text{WAIT}} \cdot \sum_{p \in P_{t'}} \sum_{rd_p \leq t \leq H'} (t - rd_p) \cdot x_{pt} \tag{4.33}$$

Subject to:

$$\sum_{rd_p \leq t \leq H'} x_{pt} = 1 \qquad \forall p \in P_{t'} \tag{4.34}$$

$$\sum_{o \in O} y_{p\tau o} \geq x_{pt} \qquad \forall p \in P_{t'}, rd_p \leq t \leq H', \tau = t + preop_p \tag{4.35}$$

$$\sum_{p \in P_{t'}} \xi_p^k \cdot y_{pto} + \sum_{\substack{p \in A_{t'} \\ t = a_p + preop_p}} \xi_p^k - CAP_{ot} \leq ot_{ot}^{o}(\xi^k) \qquad \forall o \in O, t' \leq t \leq H',$$

$$k = 1, \ldots, K \tag{4.36}$$

$$CAP_{ot} - \sum_{p \in P_{t'}} \xi_p^k \cdot y_{pto} - \sum_{\substack{p \in A_{t'} \\ t = a_p + preop_p}} \xi_p^k \leq ot_{ot}^{u}(\xi^k) \qquad \forall o \in O, t' \leq t \leq H',$$

$$k = 1, \ldots, K \tag{4.37}$$

$$\sum_{\substack{p \in A_{t'} \\ a_p \leq t < a_p + \zeta_p^k}} 1 + \sum^{p \in P_{t'}} \sum_{rd_p \leq \tau \leq t < \tau + \zeta_p^k} x_{p\tau} \leq BEDS + M \cdot z_{tk} \qquad \forall t' \leq t \leq H', k = 1, \dots, K \quad (4.38)$$

$$\sum_{k=1}^{K} z_{tk} \leq \lfloor (1 - \eta) \cdot K \rfloor \qquad \forall t' \leq t \leq H' \quad (4.39)$$

$$x_{pt} = 0 \qquad \forall p \in P_{t'}, t' \leq t \leq rd_p \quad (4.40)$$

$$y_{pto} = 0 \qquad \forall p \in P_{t'}, o \in O, t' \leq t \leq rd_p + preop_p \quad (4.41)$$

Expressions (4.33) - (4.37) are all straightforward conversions of their counterparts in model (4.16)-(4.26) that consider a limited set of realizations ξ^k.

To model the chance-constrained bed usage, the additional aid variables z_{tk} are introduced in constraints (4.38) to indicate whether the bed capacity $BEDS$ has been exceeded in sample k at time t, by means of a Big M formulation (with M a sufficiently large constant, for example $|P|$). Expression (4.39) then constrains, for any day t of the scheduling horizon, the portion of samples in which bed usage exceeds capacity (i.e. $z_{tk} = 1$) to be less than or equal to $\lfloor (1 - \eta) \cdot K \rfloor$, i.e. the proportion of samples that may exceed capacity at certainty level η.

Also note that for patients $p \in A_{t'}$ for which $a_p < t'$ (i.e. patients that are currently admitted to a hospital bed), ζ_p^k is sampled from ζ_p conditioned on the event $\zeta_p > t' - a_p$. Therefore, a conditional sampling approach must be employed for patients that are already admitted.

4.3.3 Local search

Preliminary testing with the SAA model described in the previous section showed that the model was computationally too slow for current integer programming solvers, even with small size instances. Therefore, a local search-based metaheuristic was implemented to solve the model.

A Late Acceptance Hill Climbing (LAHC) procedure was developed to solve SAA model (4.33)-(4.41). Late Acceptance Hill Climbing [9] is a simple but effective, list-based threshold accepting metaheuristic. It has been successfully applied to several optimization problems such as time-tabling [9], lock scheduling [80] and the ROADEF/EURO Challenge 2012 (Google machine reassignment problem) [31, 77].

A general pseudocode is presented in Algorithm 4. The algorithm requires only

Algorithm 4 Pseudocode for the LAHC metaheuristic

```
Require: L, s_0, f : s ↦ ℝ
  s* ← s_0, s ← s_0                     ▷ s*, s maintain best found/current solution
  laList ← (f(s*), f(s*), ..., f(s*))   [L entries]
  i ← 0
  while termination criterion not met do
      N ← SelectNeighbourhood()
      s' ← N(s)                         ▷ Sample a neighbouring solution of s
      if f(s') ≤ f(s) or f(s') ≤ laList[i mod L] then
          s ← s'
          if f(s') < f(s*) then
              s* ← s'
          end if
      end if
      laList[i mod L] ← f(s)
      i ← i + 1
  end while
  return s*
```

one parameter (list length L), an initial solution s_0, an objective function $f(s)$, a set of local search operators/neighbourhoods and a stopping criterion. The main idea behind the algorithm is that a candidate solution s' (obtained by applying a local search operator to the current solution s) is compared to the solution that was 'current' L iterations ago. This is implemented by maintaining a circular buffer of size L, containing objective function values $f(s)$ of the previous L current solutions. The gap between $f(s')$ and $laList[i \textbf{ mod } L]$ leaves room for diversification and escaping from local optima. However, as the current solution value $f(s)$ is constantly added to the buffer and older solution values are removed, the search is still directed in an improving direction.

Solution representation

The local search approach employs a straightforward solution representation based on decision vectors $ad = (ad_{p_1}, ad_{p_2}, \ldots, ad_{p_N})$ and $ot = (ot_{p_1}, ot_{p_2}, \ldots, ot_{p_N})$ that maintain resp. the admission date and the operating room assignment for each patient $p \in P_{t'}$. In addition, the solution representation maintains a fixed set of samples of size K for the LOS vector ζ^k and surgery duration vector ξ^k.

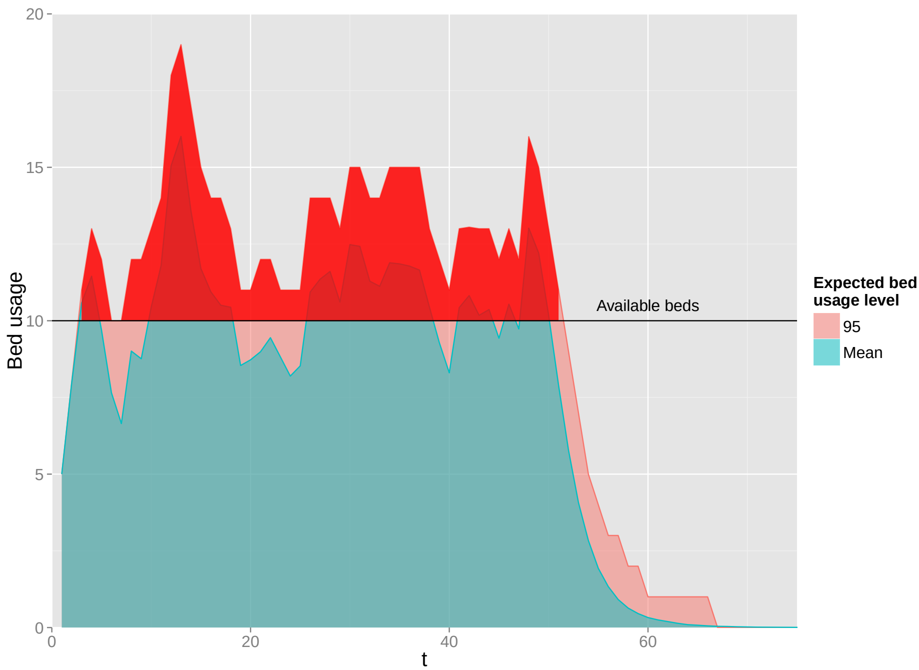


Figure 4.1: Example showing bed usage for 100 patients arriving at a rate of 2 per time unit (Poisson distributed arrivals), with log-normally distributed length of stay with mean = 5 and stdev. = 3. The local search objective function minimizes the η^{th} percentile bed shortage (red colour) with a high cost. In this example $\eta = 95\%$.

For any given assignment of *ad* and *ot* and for each sample $k = 1, \ldots, K$, the bed usage, operating theatre cost and total waiting time can easily be evaluated.

To enforce the chance-constrained bed usage constraint, the η^{th} percentile of bed usage is compared to the available capacity for each day t. The total bed shortage at the η^{th} percentile is computed (see Figure 4.1) and is penalized in the objective function with a significantly higher weight than W_{OT} and W_{WAIT}.

Local search operators

The local search employs three local search operators to perturb a solution s to a solution s'.

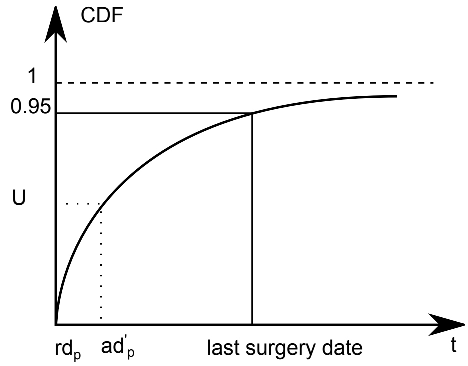


Figure 4.2: Sampling ad'_p according to an exponential distribution, with a 95% probability of sampling before the last surgery date (i.e. $\gamma_{CA} = 95\%$). U denotes a uniform random number between 0 and 1.

- Change Admission (**CA**): this local search operator samples a random patient $p \in P_{t'}$ and assigns a random new admission date $ad_p \leftarrow ad'_p$. ad'_p is sampled from an exponential distribution (with rate parameter λ) that is constructed such that γ_{CA} of all samples fall within the interval $[rd_p, lastSurgery)$, with $lastSurgery = \max_{p \in P_{t'}} \{ad_p + preop_p\}$ (see Figure 4.2 for an example with $\gamma_{CA} = 95\%$). If $ad'_p > H'$, i.e. it falls out of the scope of the current scheduling horizon, then $H' := ad'_p + \zeta_p^{MAX}$. By doing so, the sampling method is able to sample admission dates far beyond the current last planned surgery date, though at an exponentially decreasing probability. This allows the local search to postpone admissions far enough, such that meeting the bed usage constraint is ensured.

  The λ parameter can be determined by:

$$1 - e^{\lambda \cdot (lastSurgery - rd_p)} = \gamma_{CA}$$

$$1 - \gamma_{CA} = e^{\lambda \cdot (lastSurgery - rd_p)}$$

$$\ln(1 - \gamma_{CA}) = \lambda \cdot (lastSurgery - rd_p)$$

$$\lambda = \frac{\ln(1 - \gamma_{CA})}{(lastSurgery - rd_p)} \tag{4.42}$$

  Additionally, it is checked that ad'_p is not a weekend-day, as in practice generally no elective admissions are planned in weekends. If that is the case, the sample is rejected.

- Change OR (**CO**): this local search operator samples a random patient $p \in P_{t'}$ and assigns a random new operating room $o' \in O$ (if $|O| > 1$). This operator is not used if $|O| = 1$.
- Swap admissions (**SA**): this local search operator randomly selects two patients $p_1, p_2 \in P_{t'}$ and swaps their admission dates (if feasible with respect to rd_{p_1}, rd_{p_2}).
- Move OT Block (**MOB**): this local search operator samples a non-empty OR and day (i.e. it contains planned surgeries) and moves all related admissions to an earlier, empty OR and day in the scheduling horizon. This local search operator was added to compact schedules as much as possible (thus minimizing waiting time) and to speed up convergence of the algorithm.

4.3.4 Admission strategies

Using the SAA model (4.33)-(4.41) and the local search approach, four different admission scheduling strategies have been developed that vary on two different parameters: the frequency of decision making and the scheduling window.

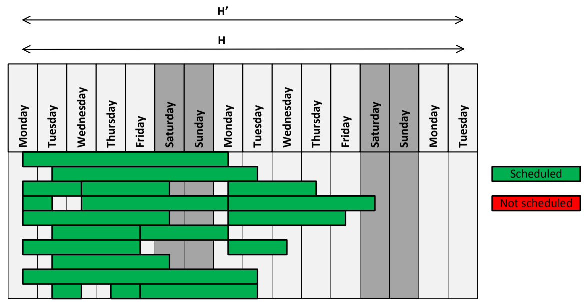
- Frequency of decision making: the frequency of decision making is a factor that determines how long patients are accumulated on a waiting list until a new admission schedule is constructed for the upcoming scheduling period. In this work, two frequency levels are considered: daily and weekly.
  - In the daily setting, every day a new admission schedule is constructed to consider the patients that requested to be admitted on that day. For example, consider that on a Tuesday three requests for admission were added to the waiting list. On Tuesday eve, a new admission schedule for Wednesday (and onwards, see scheduling window) would be constructed that considers these three admission requests and any other request still in the waiting list.
  - In the weekly setting, the admission schedule is constructed only once a week (e.g. on Friday) considering any admission requests that were made in the past week and any other admission request still in the waiting list.
- Scheduling window: the scheduling window H determines how far into the future admission dates are fixed. Note that SAA formulation (4.33)-(4.41) plans all patients in the waiting list over a period H' (which is an upper bound to plan all patients). H on the other hand determines

that portion of the planned admissions that is effectively fixed. If only a portion of admission requests are fixed, more flexibility is given to upcoming scheduling periods to achieve higher efficiency. However, this also means that some admission requests may stay in the waiting list for several iterations, ultimately increasing their waiting time.

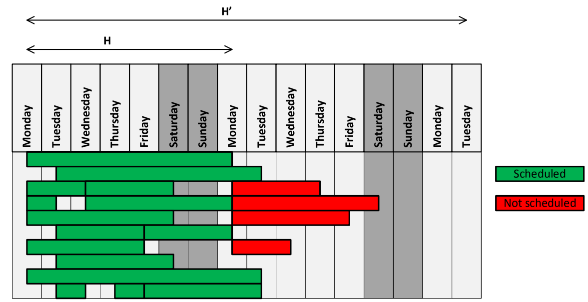
- If $H = H'$, then all admissions that are scheduled, are effectively fixed (i.e. a_p is set for all patients in the waiting list) and removed from the waiting list. Fixed admission dates are never changed at subsequent decision times.
- If $H = 1$, then only admissions planned on the first day of the upcoming scheduling period are fixed. This leaves all other planned admissions beyond $H = 1$ on the waiting list. Obviously, for week scheduling (see above), this does not make sense. Only one day would be planned, leaving all other days open. Therefore, for daily scheduling $H = 1$ is used, whereas for weekly scheduling $H = 7$ is used. In this case, daily admission scheduling with $H = 1$ will only fix admission dates for the upcoming day, whereas weekly admission scheduling with $H = 7$ will only fix admission dates for the upcoming week.

Figure 4.3 gives an overview of these four different strategies. In the weekly scheduling strategies (Figures 4.3a and 4.3b), the first day of the scheduling period is always a Monday (assuming elective admissions are planned only on weekdays, which is common practice). If $H = H'$ (Figure 4.3a), then all planned admissions are effectively fixed. If $H = 7$ (Figure 4.3b), then only admissions planned in the upcoming week are fixed. In the daily scheduling strategies (Figures 4.3c and 4.3d), the first day of the scheduling period can be any day from Monday to Friday. Again, if $H = H'$ (Figure 4.3c), then all planned admissions are effectively fixed. If $H = 1$ (Figure 4.3d), then only admissions for the upcoming day are fixed. Algorithm 5 presents pseudo code formally describing the strategies.

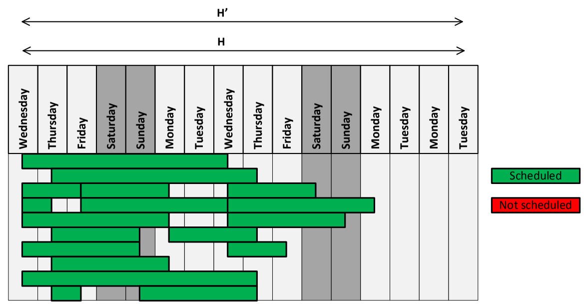
This last admission strategy (daily, $H = 1$) is clearly the least patient-friendly, as patients can only be notified one day before admission. The daily admission strategy with $H = H'$ is most patient-friendly, as patients are given their admission date on the day of their request, although it must be noted that this can be short notice (if some capacity is still available in the upcoming day). The weekly admission strategies are milder in this sense than their daily counterparts, as either a patient is given an admission date at the end of the week in which the admission request was made (if $H = H'$), or the admission date is given the week before the patient is admitted.



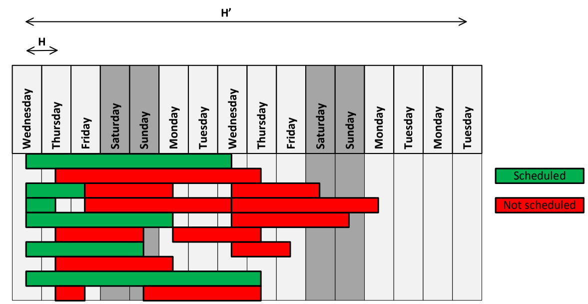
(a) Week scheduling fixing all admissions.



(b) Week scheduling fixing only current week.



(c) Daily scheduling fixing all admissions.



(d) Daily scheduling fixing only current day.

Figure 4.3: Illustration of four different admission strategies, visualizing patient admissions with expected LOS. H' denotes an upper bound on the scheduling horizon required to schedule all patients. H denotes the interval in which admission dates are effectively fixed. Green coloured patient admissions have an admission date $ad_p \in [t', H)$ and are fixed after the optimization algorithm has finished. Red coloured patient admissions having $ad_p \notin [t', H)$ are not fixed.

Algorithm 5 Pseudocode: Application of the different admission strategies in an online setting.

Require: P, $freq \in \{daily, weekly\}$, $H \in \{H', 1/7\}$
  $t' \leftarrow 0$
  $A_{t'} \leftarrow \emptyset$
  $P_{t'} \leftarrow \emptyset$
  while $P \neq \emptyset$ **do**
    $P_{t'} \leftarrow P_{t'-1} \cup \{p \in P : r_p = t'\}$
    $P \leftarrow P \backslash \{p \in P : r_p = t'\}$
    if $freq = daily$ and (t' **mod** $7 \neq 5 \wedge t'$ **mod** $7 \neq 6$) **then** ▷ Solve every weekday
      Solve (4.33)-(4.41) starting from next day (Monday if t' **mod** $7 = 4$).
    else if $freq = weekly$ and t' **mod** $7 = 4$ **then** ▷ Solve every Friday
      Solve (4.33)-(4.41) starting from Monday.
    end if
    $A_{t'+1} \leftarrow A_{t'} \cup \{p \in P'_t | a_p < H\}$
    $P_{t'} \leftarrow P_{t'} \backslash \{p \in P'_t | a_p < H\}$
    $t' \leftarrow t' + 1$
  end while

4.4 Computational study

4.4.1 Experimental setup

Experimental data

The university hospital of Leuven, UZ Leuven (UZL), has provided a dataset of hospital admissions and surgery plans. The hospital comprises 1995 beds spread over four campuses and is one of the largest hospitals in Belgium.

The dataset spans the year 2013, during which 62871 patient have been admitted. From this set 33663 admissions having a surgical pathway were considered. The following relevant data on surgical admissions was extracted: surgical disciplines, surgical procedures per discipline, surgical durations and length of stay per surgical procedure. For each surgical procedure of each discipline a log-normal distribution was fitted to both the surgical duration and the LOS. The log-normal distribution has been shown to fit well for both the surgical duration [70] as the LOS [37, 52] (although for the latter, alternative positive skewed distributions have been proposed as well, e.g. Exponential [33], Weibull [52], Phase-type [25]). Figure 4.4 and 4.5 show for example, log-normal fittings of both the surgery duration and length of stay for abdominal surgery (ABD) and thoracic surgery (THO). Some general information on the characteristics of this data and the corresponding fitted distributions is presented in Table 4.1.

A dataset of test instances was generated using the following method. Assume a given surgical discipline s, a fixed number of operating rooms $\#OT$ and a fixed number of patients $\#P$. Let $j \in J_s$ denote the surgical procedures of surgical discipline s and $f_s(j)$ the relative frequency/occurrence of surgical procedure j. Let ξ_{sj}, ζ_{sj} denote the surgical duration resp. LOS distributions of procedure j:

1. $\#P$ patient admission requests are generated according to a Poisson arrival process. The mean arrival rate is determined by $\lambda := \frac{\#OT*CAP_{OT}}{E[\xi_s]}$, with $E[\xi_s]$ the expected value of the surgical durations for surgical discipline s.

   CAP_{OT} is set to 480 min., i.e. operating rooms are staffed for 8 hours. Furthermore $CAP_{ot} := CAP_{OT}$ for all $o \in O, t = 1, \ldots, H'$, except for weekend days, where $CAP_{ot} := 0$. Thus, operating capacity is assumed to be constant during weekdays and operating rooms are closed for elective surgeries during the weekend. Furthermore, no patient admission requests are generated during weekend days.

2. For each patient admission request, a surgical procedure j is randomly selected from J_s according to the relative frequencies $f_s(j)$ of these procedures. The corresponding surgical duration and LOS distributions are associated with the patient request, i.e. $\xi_p := \xi_{sj}$ and $\zeta_p := \zeta_{sj}$.

3. de Bruin et al. [19] show that the Erlang loss queueing model with general service time distribution (also denoted M/G/c/c model in Kendall's notation [44]) is a good fit for dimensioning hospital wards to meet a certain blocking (cancellation) probability/risk. The model is therefore useful to determine bed capacity for the test instances, under different blocking probabilities.

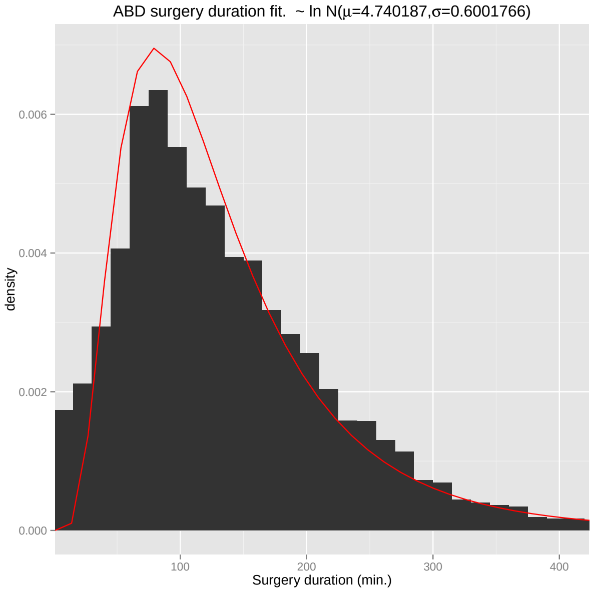
   Let μ denote the mean length of stay for surgical discipline s and let λ denote the mean arrival rate. The Erlang loss formula

   $$P_b = \frac{(\lambda\mu)^b/b!}{\sum_{k=0}^{b}(\lambda\mu)^k/k!} \tag{4.43}$$

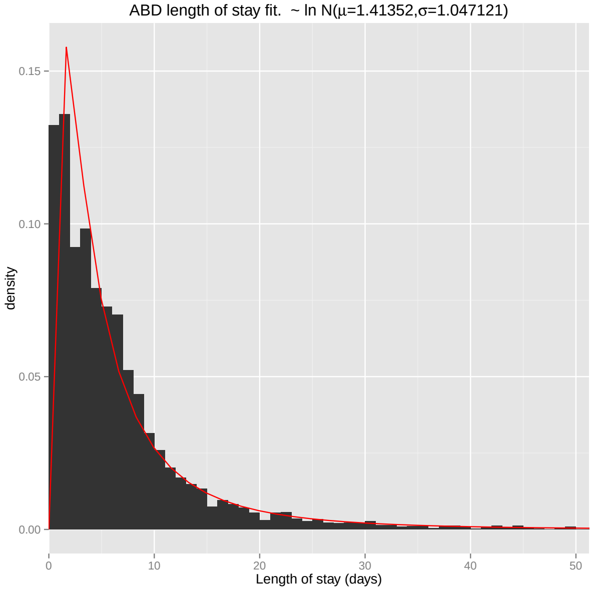
   determines the probability P_b of blocking (i.e. a bed shortage) in a hospital ward with b beds.

   Given a fixed blocking probability P_b^*, the required number of beds to meet that probability can be found by enumeration on b until $P_b \leq P_b^*$.

Instances were generated according to the following combination of considered factors:

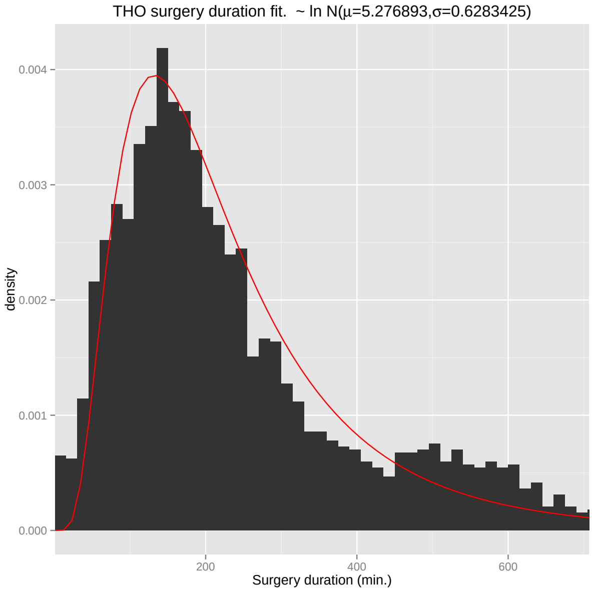


(a)

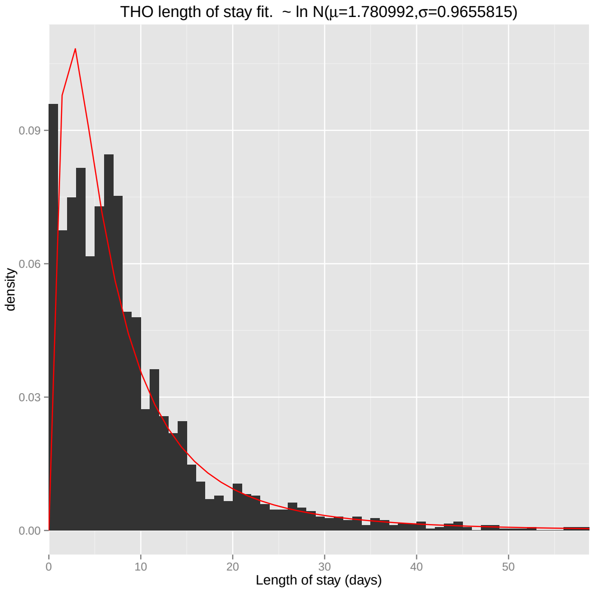


(b)

Figure 4.4: Log-normal fitting of surgery duration (a) and length of stay (b) over all abdominal surgery (ABD) procedures.



(a)



(b)

Figure 4.5: Log-normal fitting of surgery duration (a) and length of stay (b) over all thoracic surgery (THO) procedures.

Discipline	rel. frequency (%)	Surg. Duration (min.)		LOS (days)		# unique procedure types[a]
		Mean	Stdev.	Mean	Stdev.	
Abdominal surg. (ABD)	19.8	137.04	90.24	7.11	10.04	105
Cardiac surg. (CAH)	8.3	292.70	128.54	11.97	14.83	39
Gynaecology (GYN)	3.6	189.42	109.97	3.10	1.84	27
Neurosurg. (NCH)	7.6	223.47	136.66	10.13	24.55	49
Otorhinolaryngology (NKO)	5.4	173.98	106.04	4.99	10.64	35
Oncology (ONC)	5.4	106.13	86.93	7.64	15.71	30
Thoracic surg. (TH0)	7.6	238.48	165.93	9.46	11.74	60
Abdominal transplant surg. (TRA)	2.2	264.31	181.74	12.97	12.18	9
Traumatology (TRH)	13.4	141.30	79.48	7.02	12.34	84
Urology (URO)	10.4	109.88	83.48	4.58	7.80	66
Vascular surg. (VAT)	8.2	125.66	102.54	4.93	10.43	39
Plastic, reconstructive and cosmetic surg. (RHK)	4.5	195.10	171.94	7.67	14.36	27
Oral and maxillofacial surg. (MKA)	2.0	236.95	197.21	4.24	13.23	13
Orthopaedic surg. (ORT)	1.8	218.35	179.00	11.62	17.02	11

Table 4.1: Characteristics of log-normally fitted UZL surgical data of 2013.

[a] Unique procedure types with > 10 surgeries performed in 2013

Factor	Considered values
Instance size $(\#OT, \#p)$	$(1,100); (2,200); (4,400)$
Surgical disciplines	ABD,GYN,NCH,NKO,ONC,THO,ORT, TRA,TRH,URO,VAT,RHK,MKA
Blocking factor P_b^*	1%, 5%, 10%

Table 4.2: Instance generation values for the different factors under consideration. Refer to Table 4.1 for discipline abbreviations.

- Instance size: instance size is determined by $\#P$ and $\#OT$. Clearly $\#P$ influences instance size as the overall scheduling horizon will increase, whilst $\#OT$ increases scheduling complexity as more operating rooms can be used.
- Surgical discipline: different disciplines have different characteristics with respect to mean and variation of surgical procedure durations and LOS.
- Blocking factor: the probability of bed shortages is a crucial factor determining the 'bottleneck' for admission scheduling. Low blocking probability will shift the bottleneck towards the operating theatre capacity, whereas a higher blocking probability will shift the bottleneck towards ward capacity.

Table 4.2 gives an overview of the values that were considered for these factors. A full factorial set of instances was generated, with 5 different instances (generated using a different seed) per combination resulting in 585 instances in the dataset. Note that cardiac surgery (CAH) was omitted. CAH is less interesting as it has a very high mean surgical duration (292.70 min.), resulting in only one case being scheduled per OR per day. This leaves no room for optimization.

Admission strategies

The four different admission strategies discussed in Section 4.3.4 were applied to all instances using the pseudo code presented in Algorithm 5. The following parameters were determined in preliminary testing to configure the local search algorithm and the SAA model:

- sample size K: a sample value of 100 was determined to give a good balance between statistical accuracy of the SAA model and the speed of the local search algorithm (iterations per second). Figure 4.6 presents the relative deviation of the OT cost, OT overtime, OT undertime and the bed shortages from their true expected value, as a function of the sample size K. Higher values (above 100) will slightly increase statistical accuracy, but slow down the local search algorithm (as much more constraints/averages must be evaluated) further. For this experimental setup, the aim was an average execution time of at most 10 minutes (per execution of the local search algorithm) in order to keep total experimentation time manageable.

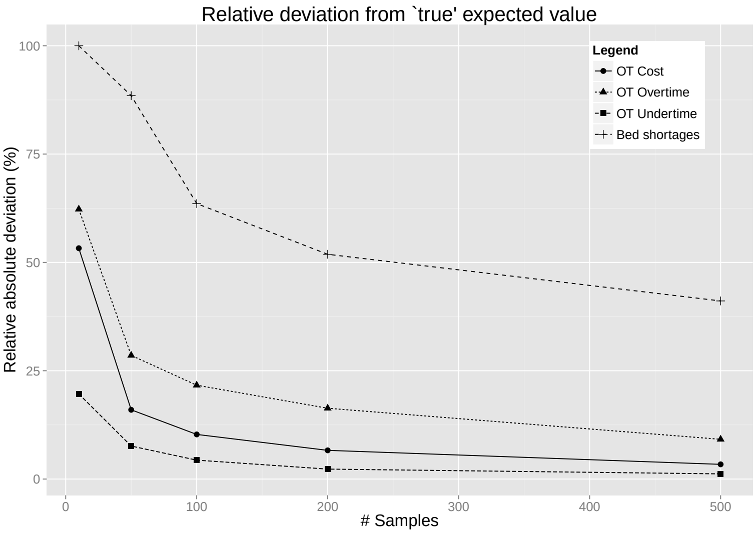
In addition, a sample size of '1' was also used, using the distributions expected value for both the surgical duration and the LOS. The benefit of the stochastic model over a deterministic model (with expected values) can thus be determined by comparing to the $K = 1$ approach.

- LA list length L: a list length of 500 was determined to give a good balance between early convergence and local search duration. Figure 4.7 shows the influence of the L parameter on the OT cost after optimization. Clearly, the L parameter influences the quality of the solution found after optimization: higher values of L allow for more diversification and ultimately result in better solutions.
- Timeout criterion: the local search algorithm assumes convergence after 25000 non-improving solutions were sampled and thus terminates.
- In this experimental setup, we have assumed that hospital management wants to meet bed availability at a certainty level $\eta = 95\%$.

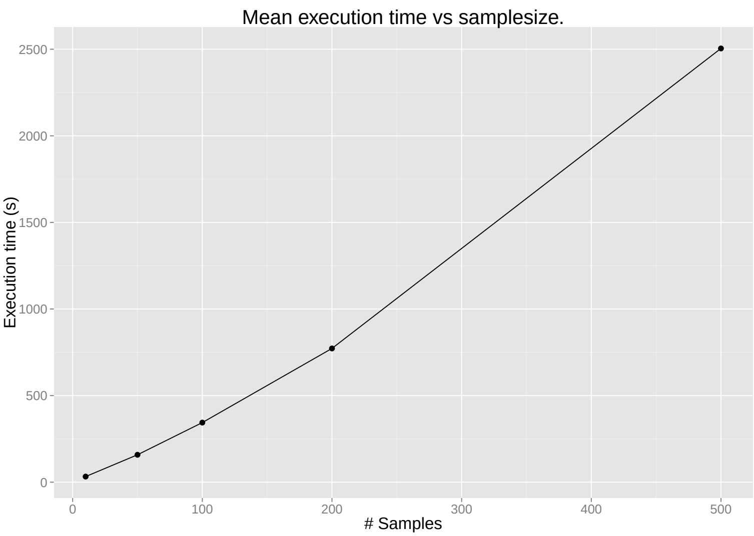
In addition to the four admission strategies, tested with both $K = 1$ and $K = 100$, four 'real-time' baseline admission strategies were also tested for comparison.

- First Fit OT (FFOT): when a new request for admission arrives, the first OR slot that has sufficient capacity to fit the mean surgical duration is assigned.
- Best Fit OT (BFOT): when a new request for admission arrives, the best OR slot that has sufficient capacity and minimal slack to fit the mean surgical duration is assigned.
- FFOT-Bed and BFOT-Bed: the counter parts of the previous two strategies that additionally check whether or not there is sufficient bed capacity for the mean LOS of the admission request.

In particular, FF OT is a good base for comparison to practice, where surgery admission dates are often assigned to the first available slot and under the assumption that a bed will be available.

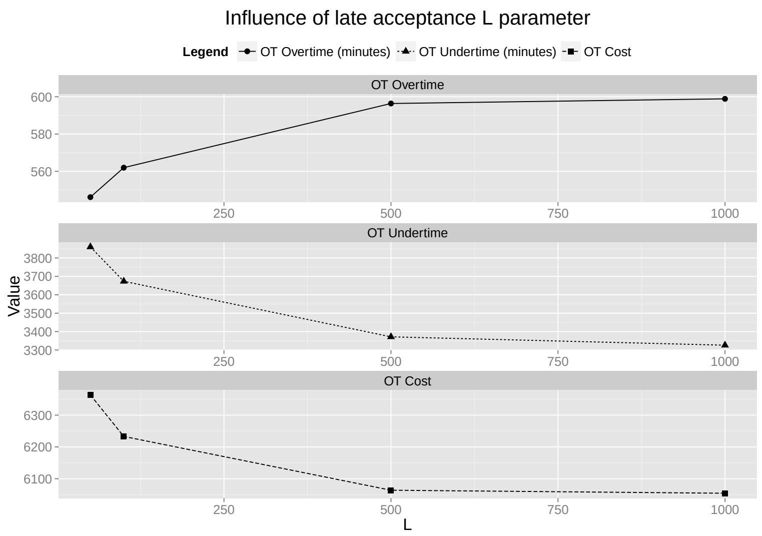


(a)

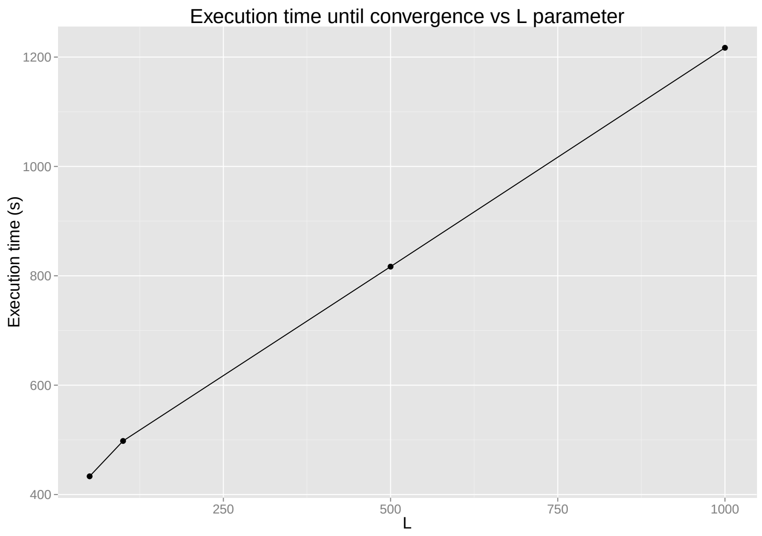


(b)

Figure 4.6: (a) Accuracy of the average OT cost, OT overtime, OT undertime and bed shortages as a function of the number of samples, relative to their 'true' expected value as determined by 10000 samples. The values represent average results obtained over a subset of the test instances (all ABD instances, 45 in total). (b) Execution time as a function of the number of samples.



(a)



(b)

Figure 4.7: (a) Influence of the L parameter on the expected OT cost after optimization. The values represent average results obtained over a subset of the test instances (all ABD instances, 45 in total). (b) Execution time until convergence as function of L.

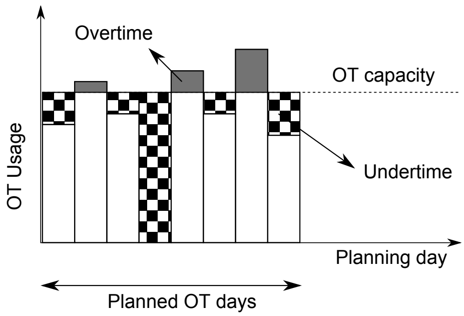


Figure 4.8: Operating theatre performance measures for a single OR. For multiple ORs, total overtime and undertime is measured over all ORs.

Evaluation

All solutions of the different scheduling strategies were evaluated with respect to three performance criteria: # bed shortages, OT cost and the mean waiting time for admission.

As different admissions strategies may plan surgeries over a different timespan, a relative measure is required to gain insight in performance between strategies. Therefore, to compare OT cost, the total overtime/undertime is measured between the first surgery day (day of first planned surgery) and the last surgery day, from which the total OT cost can be calculated. This cost is then divided by the planned number of surgery days to get an average daily OT cost. I.e.:

$$\text{Daily OT Cost} = \frac{\alpha \cdot \text{ Total overtime } + \text{ Total undertime}}{\#\text{Planned surgery days}} \tag{4.44}$$

Figure 4.8 illustrates these performance measures. The overtime to undertime ratio was set to $\alpha = 2$, under the assumption that overtime working hours are paid at a 100% additional rate.

To obtain an accurate estimate of the expected values of the number of bed shortages, the OT overtime, undertime and cost, 10000 evaluations were done on the final solutions obtained using samples drawn from ξ_p and ζ_p for each patient.

Concerning the relative weights between objectives, in this study it is considered that bed shortages have higher importance than OT cost and average waiting time, as bed shortages may ultimately cause cancellations and thus will also increase OT cost. Finally, hospitals are assumed to be profit maximizers and thus OT cost will take precedence over waiting time. Therefore, relative weights for the local search objective have been set such that $W_{BED} \gg W_{OT} \gg W_{WAIT}$.

Implementation and computational setup

The admission scheduling approach and all supporting code has been implemented in Java 1.8 and makes use of Apache Commons Math 3.3 for stochastic sampling routines. All tests have been performed on a workstation computer equipped with two eight-core Intel Xeon 2650 v2 2.6 GHz processors and 128 GB of main memory (RAM), running a Linux-based operating system. Only one processing thread is used per test and therefore this system was used to perform up to 16 tests in parallel (limiting available memory to 8 GB for each test).

4.4.2 Results and discussion

Tables 4.3, 4.4 and 4.5 report on the average performance results of the admission strategies at a blocking level P_b of resp. 1%, 5% and 10%. Reported are: the mean and 95$^{\text{th}}$-percentile of the relative daily bed shortage (relative to the available capacity) and the average bed occupancy, the mean daily OT overtime, undertime and cost, as well as the mean operating room utilization and the number of planned OT days (refer to Figure 4.8), the mean waiting time per patient and the execution time per instance.

At a blocking level of 1% (Table 4.3), the number of bed shortages is low for all admission strategies, as would be expected. Differences can be seen in OT cost between real-time, daily and weekly scheduling strategies.

- Real-time strategies: FFOT and BFOT perform mostly similar, having near equal OT costs and utilization. However, as BFOT minimizes slack in each OR, it will plan closer to the available capacity. This results in a higher OT overtime and lower OT undertime. Since the decrease in OT undertime does not compensate for the increase in OT overtime, this results in a (slightly) higher average daily cost. In addition, BFOT is more likely to postpone an admission slightly to minimize slack, therefore mean waiting time for patients is increased. A similar observation can be made for their variants considering bed availability. However, OT utilization is slightly lower due to postponing a fraction of surgeries because of lack of bed availability.
- Daily optimization: a distinction can be made between the stochastic approaches ($K = 100$) and the average value approaches ($K = 1$). The average value approaches plan much closer to the available capacity and reach higher utilization. However, as they have no notion of the variance, the OT overtime cost is underestimated. This results in an increased

daily OT overtime. The stochastic approaches on the other hand do have information on the variance and in the case where there is also sufficient flexibility ($H = 1$), a significant decrease in daily OT cost can be identified. Finally, the least flexible approaches with $H = H'$ perform worst. However, the stochastic approach $K = 100, H = H'$ is still able to match the daily OT cost of real-time strategies, at a higher utilization rate. Thus, mean patient waiting time can be reduced while maintaining a similar cost level.

- Weekly optimization: the same distinction between stochastic approaches ($K = 100$) and the average value approaches ($K = 1$) can be made for weekly planning approaches. Average value approaches underestimate OT overtime cost and plan at a higher occupancy. The stochastic approaches on the other hand correctly estimate the expected OT cost. In addition, OT cost is lower for the weekly optimization approaches than for the daily optimization approaches due to increased scheduling flexibility. However, this is at the expense of mean patient waiting time which is higher due to postponing decision making. Again, the most flexible, stochastic approach ($K = 100$, $H = 7$) performs best with respect to daily OT costs.

As expected, week scheduling strategies incur the highest mean waiting time per patient due to accumulating more patients on the waiting list before decision making is performed. Interestingly, admission strategies that fix all admission dates ($H = H'$) have slightly higher mean waiting time than strategies that do not fix all planned patients ($H = 1$, $H = 7$). This can be explained by the fact that in general all optimization based strategies will have a tendency to plan short surgical procedures with shorter LOS at earlier times and longer surgical procedures with longer LOS at later times. This results in decreased average waiting time and increases throughput at the start of the scheduling horizon. This effect is less pronounced when all admission dates are fixed after scheduling them, as longer procedures will not be postponed indefinitely.

As the blocking probability increases and the bottleneck shifts towards bed capacity (Tables 4.4 and 4.5), it can be observed that FFOT and BFOT, which do not consider bed usage at all, will incur non-negligible bed shortages. These bed shortages denote the average daily relative bed shortage. Bed occupancy will generally increase during the weekdays due to new admissions and drops during the weekend (since no admissions occur). Therefore, the peak bed shortages during the week may be considerable.

Non-stochastic approaches considering the expected LOS (FFOTBed, BFOTBed and admission strategies with $K = 1$) also underestimate the expected bed usage. This effect is worse for the average value daily and weekly admission strategies (i.e. with $K = 1$) that plan closer to the available bed and OT capacity as they

	Strategy	Rel. bed short. ($\frac{\%}{day}$)		Bed	OT performance (daily)				OT	Mean wait.	Exec. time
		Mean	95^{th}-p	util. (%)	OTO	OTU	cost	util. (%)	days	(days/pat.)	(ms.)
Real-time	FFOT	0.0	0.1	56.2	31.94	78.27	142.16	90.3	98.57	9.88	27.60
	FFOTBed	0.0	0.0	56.1	31.74	79.25	142.74	90.1	98.86	9.94	13.98
	BFOT	0.0	0.1	56.3	32.48	77.85	142.80	90.5	98.34	10.01	28.42
	BFOTBed	0.0	0.0	56.1	32.26	79.04	143.55	90.3	98.70	10.07	27.86
Daily	$K = 1.\ H = 1$	0.0	0.1	60.1	43.72	56.01	143.45	97.4	90.15	7.04	31182.08
	$K = 1.\ H = H'$	0.0	0.1	59.6	43.67	57.70	145.05	97.1	90.42	8.00	110279.90
	$K = 100.\ H = 1$	0.0	0.0	57.4	33.05	72.76	138.86	91.7	95.77	8.37	2163438.00
	$K = 100.\ H = H'$	0.0	0.0	58.0	37.42	68.41	143.25	93.5	93.64	8.95	720638.70
Weekly	$K = 1.\ H = 7$	0.0	0.1	60.9	44.23	54.19	142.65	97.9	89.93	12.19	9625.79
	$K = 1.\ H = H'$	0.0	0.1	60.6	44.44	54.34	143.22	97.9	89.81	13.07	6779.82
	$K = 100.\ H = 7$	0.0	0.0	58.1	33.11	71.22	137.44	92.1	95.46	13.65	758406.90
	$K = 100.\ H = H'$	0.0	0.0	58.2	34.90	68.57	138.38	93.0	94.31	14.43	465510.00

Table 4.3: Average performance results for the different admission strategies, for instances with blocking probability $P_b^* = 1\%$. Notation: OTO = OT overtime, OTU = OT undertime.

have no notion of the variance. This results in a higher utilization and admission rate, ultimately resulting in higher bed load and thus a higher probability for bed shortages.

The stochastic approaches ($K = 100$) have a better estimate of the expected bed usage and its variation and are able to reduce the risk of bed shortages. Obviously they cannot completely avoid bed shortages at the 95th percentile, as the sampling approximation is limited to 100 samples and thus may still underestimate the 95th percentile. Nevertheless, bed shortages at the 95 percentile are kept at an acceptable level. The most flexible stochastic approaches ($H = 1$ and $H = 7$) obtain both the lowest risk on bed shortages and have low OT cost. However, this is at the expense of patient friendliness (i.e. notification of admission date either the day before, or in the previous week) and increased mean patient waiting time (for week scheduling).

Interestingly, a stochastic week scheduling approach where admission dates are fixed ($H = H'$) provides a nice balance: relatively quick notification of the admission date (at the end of the week in which the patient admission request was made), reduced risk of bed shortages and a daily OT cost lower than real-time scheduling strategies that also consider bed availability.

	Strategy	Rel. bed short. ($\frac{\%}{day}$)		Bed	OT performance (daily)				OT	Mean wait.	Exec. time
		Mean	95^{th}-p	util. (%)	OTO	OTU	cost	util. (%)	days	(days/pat.)	(ms.)
Real-time	FFOT	0.3	0.8	65.2	32.05	78.81	142.90	90.3	99.09	9.94	25.43
	FFOTBed	0.1	0.4	63.9	30.73	86.77	148.23	88.3	101.36	10.44	14.10
	BFOT	0.3	0.9	65.3	32.60	78.14	143.35	90.5	98.79	10.09	27.67
	BFOTBed	0.1	0.4	63.9	31.24	86.53	149.00	88.5	101.18	10.59	25.10
Daily	$K = 1.\ H = 1$	0.2	0.8	68.5	43.01	58.83	144.84	96.7	91.32	7.41	35663.21
	$K = 1.\ H = H'$	0.2	0.6	67.1	40.88	70.79	152.54	93.8	124.19	9.99	219687.30
	$K = 100.\ H = 1$	0.0	0.2	64.9	31.83	77.00	140.66	90.6	97.51	8.79	2743409.00
	$K = 100.\ H = H'$	0.1	0.3	65.6	35.18	78.40	148.75	91.0	97.24	9.54	845337.20
Weekly	$K = 1.\ H = 7$	0.2	0.8	69.6	44.05	55.38	143.47	97.6	90.61	12.38	11267.56
	$K = 1.\ H = H'$	0.2	0.8	69.2	42.99	60.36	146.34	96.4	91.87	13.49	7785.38
	$K = 100.\ H = 7$	0.0	0.2	65.8	32.29	74.33	138.90	91.2	96.91	13.87	927730.40
	$K = 100.\ H = H'$	0.0	0.2	66.1	33.02	75.49	141.53	91.2	97.06	14.89	573612.40

Table 4.4: Average performance results for the different admission strategies, for instances with blocking probability $P_b^* = 5\%$. Notation: OTO = OT overtime, OTU = OT undertime.

	Strategy	Rel. bed short. ($\frac{\%}{day}$)		Bed	OT performance (daily)				OT	Mean wait.	Exec. time
		Mean	95^{th}-p	util. (%)	OTO	OTU	cost	util. (%)	days	(days/pat.)	(ms.)
Real-time	FFOT	1.2	2.7	72.7	31.94	78.04	141.91	90.4	98.58	9.99	26.38
	FFOTBed	0.3	0.9	69.1	28.79	98.73	156.31	85.4	104.32	11.29	14.83
	BFOT	1.2	2.7	72.8	32.45	77.43	142.34	90.6	98.33	10.11	28.48
	BFOTBed	0.3	0.9	69.2	29.26	98.86	157.39	85.5	104.37	11.40	25.16
Daily	$K = 1.\ H = 1$	0.6	1.9	74.1	40.86	63.80	145.51	95.2	92.61	7.93	123060.10
	$K = 1.\ H = H'$	0.4	1.3	72.0	37.23	87.00	161.45	89.6	131.24	10.66	114062.90
	$K = 100.\ H = 1$	0.1	0.4	69.2	28.54	86.71	143.78	87.9	100.60	9.65	3529782.00
	$K = 100.\ H = H'$	0.1	0.5	69.9	30.73	96.88	158.34	86.2	102.70	10.72	939127.70
Weekly	$K = 1.\ H = 7$	0.6	1.9	75.3	41.80	60.46	144.06	96.1	91.84	12.93	13068.21
	$K = 1.\ H = H'$	0.5	1.6	74.5	39.17	74.05	152.40	92.7	95.80	14.34	8059.26
	$K = 100.\ H = 7$	0.1	0.4	70.3	28.67	83.46	140.81	88.6	99.72	14.62	1034508.00
	$K = 100.\ H = H'$	0.1	0.5	70.7	28.78	90.37	147.94	87.2	101.69	15.88	642710.30

Table 4.5: Average performance results for the different admission strategies, for instances with blocking probability $P_b^* = 10\%$. Notation: OTO = OT overtime, OTU = OT undertime.

4.5 Conclusion

In this chapter an admission scheduling approach for scheduling elective surgical patients has been presented, for which a stochastic, chance-constrained optimization model is introduced. It aims at minimizing the expected operating theatre cost (as a function of both over- and underutilization) and patient waiting time, while limiting the risk of bed shortages to a fixed certainty level. A sample average approximation to the model is proposed and is solved by a meta-heuristic approach based on Late Acceptance. Finally, different admission scheduling strategies are constructed based on this stochastic approach and are compared in a computational study.

The results indicate that a stochastic approach is appropriate in an uncertain scheduling setting, as it is able to consider the uncertainty on patients' length of stay and surgery duration, and the resulting variance in bed usages and operating theatre occupancy and costs. When the surgical ward capacity is not a bottleneck, stochastic approaches are able to reduce expected operating theatre costs. This also depends on the admission scheduling strategy that is employed: the most flexible approach will enable to reduce expected costs, however this will be at the expense of both patient friendliness (late notification of admission date) and/or increased waiting time. On the other hand, when the surgical ward capacity is a bottleneck, stochastic approaches are able to maintain a low risk of exceeding bed capacity. However, this is again at the expense of patient friendliness and patient waiting time.

In conclusion, there is *no free lunch* in a capacity constrained uncertain environment such as a hospital. Robustness against bed shortages comes at the expense of less patient-friendly scheduling techniques (waiting lists, late notification) or at the expense of increased OT costs.

Chapter 5

A heuristic approach to multi-day surgical case scheduling considering generalized resource constraints

In the final study of this dissertation, an insight is given into the scheduling process of the operating theatre. The previous chapter already touched on the fact that the operating theatre is a key hospital resource, producing significant revenue though at substantial cost. To quantify this, consider that Macario et al. [49] indicate that surgery related services can represent more than 40% of hospital costs and revenues. In addition, Jackson [42] identifies the operating theatre as an important profit center rather than a cost center. Being a profit center, the operating theatre should be run to maximize throughput, while still managing added costs from over-utilization. In this chapter, a decision support model and approach is presented for scheduling the operating theatre to achieve this goal.

A rich multi-day surgical case scheduling problem is proposed that considers generalized resource constraints and desiderata from the surgical staff. The aim is to schedule as many surgical cases in as few operating theatres as possible, within regular operating theatre opening hours and under limited resource availability. A heuristic algorithm is presented to solve this rich problem formulation, and it is tested on a set of real-world data. The results are compared to surgical plans made by manual planners.

This work has been performed in the context of IWT project 120604, a joint research project of the research group CODeS & iMinds-ITEC and a software company, Dotnext[1]. Dotnext develops healthcare applications for the operating theatre and the cardiology department, among which a software solution for operating theatre planning and scheduling. The problem description and data were provided as part of this joint research project. The final goal of this study is the inclusion of the algorithm into the company's software application. The development of the model and the heuristic approach has been done in collaboration with *Pieter Smet, KU Leuven, Department of Computer Science, CODeS & iMinds-ITEC.*

5.1 Introduction

Efficiently running an operating theatre consisting of several operating rooms (ORs) and a large surgical staff is not an easy task. It requires the coordination of many different resources, both human and material, to be able to perform surgeries. A great deal of careful planning and organization is thus necessary to avoid delays and to ensure high throughput. Failure to do so may require overtime by the surgical staff to finish all surgeries, or even cause cancelling/postponing surgeries, ultimately resulting in loss of revenue, increase of costs and worse quality of care.

Over the past decades, numerous software development efforts have been made to assist the planner (e.g. a surgeon, an OT manager) to plan and schedule the OT. These efforts have mainly computerized the process that human planners perform, e.g. providing digital user interfaces and scheduling boards in which the manual planner can allocate surgeries. Such applications can provide the planner with a good overview of availability of resources, thus improving his/her effectiveness at the task. However, the complexity of finding a good OT schedule has not been reduced, and planners still have to 'puzzle' to find a good, workable OT schedule. Software support for generating surgical schedules automatically has seen considerably less adoption. For example, Cardoen et al. [14] report on the usage of software support for generating and optimizing surgery schedules in hospitals in Flanders, Belgium. Their survey indicates that as high as 56% of the hospitals do not use any software support to develop and optimize their surgical schedules, 26% use software support but the produced schedules are

[1] Dotnext, Dikkemeerweg 172, 1652 Alsemberg, Belgium – `http://www.dotnext.be`

not reliable (may contain errors, or do not consider all resources) and 11% use software that produces reliable/usable schedules (7% used other approaches, or left the question unanswered).

This chapter focuses on the development of a decision support model and algorithm for OT scheduling, generalizing many considerations encountered in the literature and in practice. The aim is to algorithmically support OT managers and planning offices in their daily/weekly task of both scheduling (determining start date and time of) surgeries and assigning them to an OR. In particular, the presented approach supports scheduling other resources that may be required for performing surgeries, both human resources (e.g. the surgical staff: surgeons, anaesthesiologists, instrumenting nurses) and other resources (e.g. portable imaging tools, operating lights). To this end, generalized resource requirements are introduced to define the dependency of surgeries on specific types of resources. A heuristic approach to this problem formulation is presented. It is based on a schedule generation procedure combined with local search.

The main motivation for this work was to develop an approach *general and flexible* enough to deal with different approaches to operating theatre scheduling across hospitals. This has been facilitated by the collaboration with Dotnext. Their input on current operating theatre planning practices and the provision of data has been essential. The final goal of this collaboration is to integrate the presented approach in this application to algorithmically support the planning process of the operating theatre in practice.

5.1.1 Related work

Given the central role of the OT in the operation of any hospital, and the impact it has on hospital costs and revenue, it has been the subject of a myriad of studies. In particular, optimization and decision support for planning and scheduling in operating theatres is not at all new. Cardoen et al. [13] review 115 studies on the matter published after 2000, and many more have been published since then. As also noted by Guerriero and Guido [34], a general distinction can be made between studies depending on the decision level they focus on. The surgical scheduling problem that is considered in this chapter is situated at the operational decision level. It is concerned with scheduling individual surgical cases in an operating theatre over a certain planning horizon. However, one particular decision at the tactical level that is of interest for this work is the operating theatre planning strategy. The operating theatre planning strategy determines how operating theatre capacity is distributed among different surgeons or surgeon groups. Different strategies can be identified [34]:

- *Open scheduling strategy*: in this strategy, operating theatre capacity is not reserved for surgeons or surgeon groups. Rather, the operating theatre capacity is managed as a single, shared entity in which surgical cases can be planned. Capacity is allocated on a *first-come first-served* basis. This may cause problems between surgeons and surgeon groups, as certain disciplines may be able to plan certain surgery types well in advance. Thus, they may fill up capacity before other surgeons have the opportunity to do so. Furthermore, it is more difficult to manage. For these reasons, the following strategy is more common.
- *Block scheduling strategy*: preallocates the operating theatre capacity to the different surgeons or surgeon groups. Operating theatre capacity is divided into blocks or slots consisting of an OR for a specified duration (usually either a half day/full day). A block schedule, also denoted as the master surgical schedule (MSS), determines which blocks are allocated to which disciplines for each day of the week. Surgeons are free to plan surgical cases within their allotted blocks as they see fit.

  The MSS is typically cyclical, and is repeated over the course of several weeks/months before being adjusted. By doing so, a stable and fair way of planning and working is ensured, resulting in a stable flow and consistent mix of patients. This stable situation also serves as the basis for further decision making, for example for setting staffing levels in operating theatres and the surgical wards. However, it is also one of its weaknesses. As demand for surgery changes, some blocks may become underutilized by certain disciplines while other disciplines are accumulating patients on waiting lists. Therefore, some reallocation may be necessary. This is addressed by the following strategy.
- *Modified block scheduling*: complements block scheduling with a policy to check for underutilization. If underutilization of an upcoming OR block is likely, this block may be opened to other surgeons to plan surgical cases.

As already introduced in Chapter 4, the MSS has been shown to have significant impact on other resources (bed usage, workload) as it partly determines the arrival rate and arrival pattern/variation of surgical patients. Therefore, these resources should not be ignored when constructing/updating the MSS. Beliën and Demeulemeester [5] showed how to construct an MSS that results in an expected levelled bed occupancy, by minimizing a weighted sum of the maximum expected bed occupancy and the maximum expected variance of the bed occupancy. van Essen et al. [73] also showed how to relate downstream ward bed usage to the MSS. By doing this, they were able to reduce the number of required beds in HagaZiekenhuis (Den Haag, the Netherlands) by rearranging the MSS. Next to bed occupancy, the MSS also influences the workload in nursing wards. Therefore, it is worthwhile to consider staffing decisions in the process of creating the MSS. Beliën and Demeulemeester [6] showed how to match and integrate the construction of an MSS with nurse scheduling

At the operational decision level, the focus is on scheduling individual surgical cases. Many studies further decompose the process into two steps, denoted *advance scheduling* and *allocation scheduling* [51, 60]. Other authors, e.g. [66], have coined terms such as “intervention assignment” and “intervention scheduling”, and “surgical case assignment” [1] and “surgical case scheduling” [12]. In any case, the former process deals with assigning a surgery date, and possibly an OR, to individual surgical cases in an upcoming planning period. The latter process deals with sequencing/scheduling individual surgical cases during the day within the different ORs, determining a specific start time for each surgery to be performed.

The advance scheduling problem has already been mentioned in Chapter 4 in the context of admission scheduling. Most studies however, focus solely on the operating theatre, assigning surgery dates to individual surgical cases. This is often a weekly process of selecting surgical cases from a waiting list to be performed in the upcoming week, and possibly (pre-) assigning them to individual ORs. The main objectives considered minimize overtime and underutilization of ORs [26, 43], as well as patient related costs and quality of service measures such as waiting time [35, 43] and tardiness with respect to due dates [66].

The allocation scheduling problem follows the advance scheduling problem, and takes as input the planned surgical cases for the upcoming planning period. The main goal of the planning process is to construct a feasible work plan for each surgery day. Therefore, the allocation scheduling process typically considers more resources and more operational constraints in order to be feasible in practice. Examples consider surgeons operating in multiple rooms [53, 56], the capacity-limited post anaesthetic care unit (PACU) [12, 56] or the intensive care unit (ICU), (sequence-dependent) setup times between surgeries [63, 82]. Many studies target optimization of a variety of performance measures, but mostly maximizing utilization of ORs (within restricted capacity), or minimizing overtime or makespan (while scheduling all patients) are of main importance.

Some studies tackle both the advance scheduling problem and the allocation scheduling problem in one single approach. Marques et al. [53] presented an integer programming model to both select surgical cases to be performed in the upcoming week and determine an OR and start time (slot). The model considers different surgical case urgency classes, surgeon and patient availabilities, and slack time for cleaning. Marques et al. [54] presented a genetic heuristic for the same problem that achieves even better results. Riise and Burke [66] presented an iterated local search algorithm for a combined surgery allocation and surgery admission planning problem, considering patient waiting time and tardiness, and surgeon overtime.

Another distinguishing element between studies is the consideration of uncertainty in surgery durations and emergency interventions. Denton et al. [22] presented a stochastic model for surgery sequencing and scheduling in a single OR. Their aim is to minimize a weighted objective function consisting of waiting time, idle time and tardiness. Hans et al. [36] presented a robust surgery loading study for the advance scheduling problem. Their aim is to minimize overtime and to maximize free capacity, whilst considering uncertainty in surgery durations and varying flexibility with respect to a base schedule. They minimize the required slack for avoiding overtime by exploiting the *portfolio effect* – a decrease in risk by increasing diversity (by means of non-correlated portfolio components). As already discussed in Chapter 4, Min and Yih [57] presented a stochastic approach to the advance scheduling problem that considers both uncertainty on surgery durations, emergency arrivals and length of stay in the ICU. Their objective was to minimize block overtime and patient waiting time related costs. Bruni et al. [8] presented a heuristic approach to a stochastic programming model for the advance scheduling problem considering different recourse methods. On a weekly basis, an advance schedule is constructed that maximizes an abstract priority weighted profit of performing surgeries, decreased by the expected recourse cost. Different recourse methods are considered that either perform surgical cases in overtime, redistribute surgical cases between ORs or completely reschedule.

5.1.2 Contribution

The main contribution of this study is a decision support model and approach for the surgical scheduling problem that generalizes many considerations found in literature and in practice: employing multiple ORs, assigning a surgical team, material requirements for surgical cases, etc. A generalized resource requirement model is proposed for specifying dependencies of surgical cases on different resources.

The model of Meskens et al. [56] is one that considers many aspects of the surgical case scheduling problem encountered in practice. The study has a similar goal as ours, namely to present a general decision support tool. Meskens et al. present a modular model for the daily surgical case scheduling problem using constraint programming, in which different considerations are grouped into modules and can be turned 'on/off' according to the application's requirements. However, the work presented in this chapter also accommodates several other considerations, such as minimizing resource transfers (e.g. avoiding that nurses, anaesthesiologists have to move between rooms), the concept of *surgical phases* (e.g. a surgeon should only be present during an incision phase) and multi-day scheduling.

Many studies [e.g. 12, 53, 54, 56] employ a discrete representation of time in their planning models. Both the planning horizon and the surgical case durations are discretized into multiples of a certain time unit. To keep model sizes tractable, time unit sizes are often in the order of 15-30 minutes. As surgical durations must be rounded due to this discretization underutilization may be introduced. The approach presented in this study uses a continuous time representation, enabling any precision of scheduling and avoiding underutilization due to limited precision. More important, the algorithm complexity does not depend on the precision of time.

Finally, an important practical contribution is that the approach is being prepared for implementation, as part of the joint research project with Dotnext. The generality and flexibility of the model and the approach are essential in this aspect, as the software application is employed in many different hospitals and must be able to cope with this diversity.

5.2 Surgical case scheduling problem

The surgical case scheduling problem (SCSP) deals with scheduling a set of surgical cases S in a set of ORs O, over a finite time horizon of one or more days. In this work, an *open-scheduling* policy is assumed, allowing surgical cases to be planned in any OR at any time. As also noted by Fei et al. [26], an open scheduling strategy is more general than block scheduling in the sense that schedules fitting in a block scheduling strategy, also fit an open scheduling strategy. Nevertheless, block scheduling, which is most common in practice, can also be accommodated.

The aim is to schedule as many surgical cases in as few ORs as possible, within the regular opening hours of the OT. Furthermore, a set of hard resource and ordering constraints (cannot be violated) must be considered. Soft resource constraints and desiderata are penalized if violated or not met. In what follows, the elements and constraints of the problem are described. Notation will be introduced along the description but is also summarized in Tables 5.1 and 5.2.

Notation	Description
$s \in S$ ($\lvert S\rvert = N$)	the set of surgical cases
$o \in O$	the set of ORs
$H = \{1, \ldots, \lvert H\rvert\}$	the planning horizon (in days)
$j \in RT$	the set of resource types
$r \in R$	the set of resources
$RT_r \subseteq RT$	the set of resource types that resource r can be assigned to.
$R_j \subseteq R$	the set of resources of type j. Note that R_{j_1} and R_{j_2} are not necessarily disjoint for $j_1 \neq j_2$ i.e. some resources are flexible and can fulfil different resource type requirements (e.g. nursing staff members that have multiple skills).
$R^{TRAN} \subseteq R$	the set of resources for which transfers between ORs must be minimized.
$R^{IDLE} \subseteq R$	the set of resources for which idle time must be minimized.
C_r	the number of ORs resource r can be used in.
$AF(r_1, r_2)$	the affinity (positive, negative, neutral) between resources r_1 and r_2.
$[\text{start}, \text{end}) \in AV_{o,d}$	Availability intervals for OR o on day d.
$[\text{start}, \text{end}) \in AV_{r,d}$	Availability intervals for resource r on day d.

Table 5.1: Summary of problem input sets and notation.

Notation	Description
$D_s \subseteq H$	the days on which case s can be planned
p_s	the priority of case s
d_s	the surgical duration of case s
s_{sj}	start of surgical phase (offset from start of case s in the OR) of resource type j
d_{sj}	duration of surgical phase of resource type j
$O_s^1, O_s^2, O_s^3 \subseteq O_s$	the set of preferred, possible and *if-necessary* operating rooms
$RT_s^r, RT_s^o \subseteq R$	the set of required (optional) resource types
C_{sj}	the specific number of resources of resource type j that are requested (required or optional) to be present.

Table 5.2: Summary of surgical case attributes.

5.2.1 Basic problem

The aim of this work is to schedule as many surgical cases in an OT over a specified planning period as possible. In practice it is common to schedule the surgical cases for the upcoming week, one week in advance. However, it is not uncommon that certain arrangements have already been made with respect to the surgery date of individual cases. Therefore, this work considers a general setting where surgical cases $s \in S$ are eligible to be scheduled on one day from a set of possible days $D_s \subseteq H$. Clearly, a surgical case should only be scheduled once:

Constraint 1. *A surgical case s can be scheduled on at most one day $d \in D_s$.*

It is assumed that an open scheduling policy is maintained, i.e. surgical cases can be scheduled in any OT. However, practice may differ: some surgical cases may only be performed in certain ORs, due to restrictions on capacity, fixed equipment, etc. Therefore, for each surgical case s a set of suitable ORs O_s is considered. Obviously:

Constraint 2. *A surgical case s can be scheduled in at most one OR $o \in O_s$.*

In addition, some ORs may be more suited than others for a surgical case s. Therefore, a distinction is made between ORs. "Preferred" rooms ($O_s^1 \subseteq O_s$) are best suited for performing surgery for case s, but "possible" rooms ($O_s^2 \subseteq O_s$) and "if-necessary" rooms ($O_s^3 \subseteq O_s$) may be used as well. However, "if-necessary" rooms should be avoided but can be used if no other room is available. Thus:

Soft constraint 1. *Surgical cases should be planned in "preferred" rooms as much as possible, but "possible" and "if-necessary" rooms may be used as well. "If-necessary" rooms should be avoided.*

Each surgical case s has a specific duration d_s during which the case occupies the OR. Clearly:

Constraint 3. *A surgical case s cannot be overlapping in time with any other surgical case s' scheduled in the same OR.*

ORs are only 'open' during specific time windows, often only from 7-8 am until 5-6 pm. Sometimes an OR may also be closed during lunch. Therefore, availability intervals can be specified for each OR and each day d as $[\text{open}, \text{close}) \in AV_{o,d}$, and:

Constraint 4. *A surgical case s can only be scheduled on day d in OR o within an availability interval* $[open, close) \in AV_{o,d}$.

5.2.2 Resource dependencies

Surgical procedures may require the presence of some resources when being performed. In this problem setting, resource requirements are considered in a broad sense: both human resources such as surgeons, anaesthesiologists, nurses, but also specific material such as portable imaging equipment, surgical lights or other tools.

We distinguish between resource types and resources. Resource types (denoted $j \in RT$) represent general types such as surgeons, nurses, instrumenting nurses, anaesthesiologists, lamps, imaging devices. That is, they represent a specific functionality (for materials) or role (for people). Resources (denoted $r \in R$) on the other hand represent true physical resources that have specific functionalities or can perform certain roles. The set of resource types a resource r belongs to is denoted by $RT_r \subseteq RT$, and the set of resources of a certain resource type j is denoted $R_j \subseteq R$.

Each surgical case specifies dependencies on resource types rather than specific resources, and are denoted by RT_s. For each resource type dependency $j \in RT_s$, a count C_{sj} specifies the number of resources required for a specific resource type. Therefore, an additional complexity in this model is that for each resource type dependency $j \in RT_s$ of a surgical case s, sufficient resources $r \in R_j$ must be assigned.
Furthermore, resources may only be needed during a specific part of the surgical case duration. For example, an imaging tool may only be needed at the start of a surgical case. A (supervising) surgeon on the other hand may only be present during the incision phase of the surgical case. Therefore, each resource type requirement $j \in RT_s$ a surgical case s depends on, also specifies a *surgical phase* by an offset s_{sj} from the planned start, and a duration d_{sj} (Figure 5.1). Resources only need to be assigned during this surgical phase, rather than during the entire surgical case.

As with ORs, resources also have limited availability. Thus, for each resource r availability intervals are specified for each day d of the planning period as $[\text{start}, \text{end}) \in AV_{r,d}$.

Finally, a distinction is made between required and optional resources (denoted RT_s^r and RT_s^o respectively), the former being necessary for planning the surgical

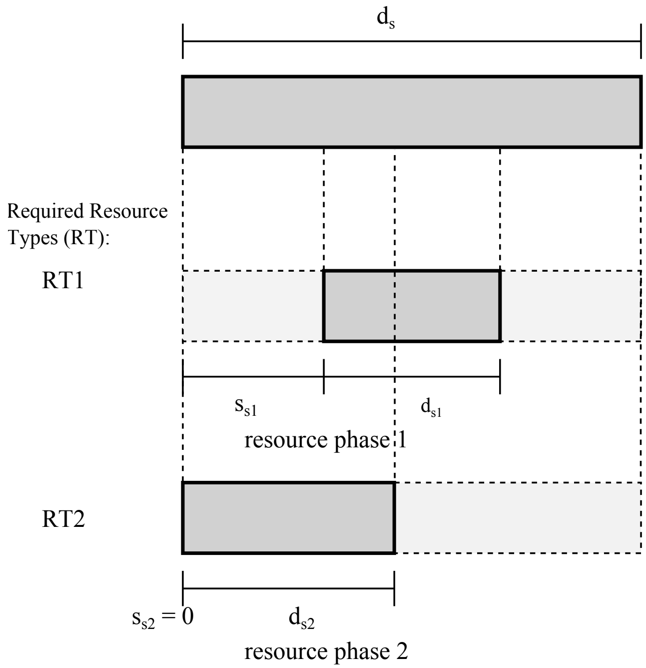


Figure 5.1: Definition of a resource phase. For surgical case s, one dependency on a resource type (RT1) is defined, with offset from start s_{s1} and duration d_{s1}.

case, while the latter are preferably present. The resource constraints are thus:

Constraint 5. *For each required resource type $j \in RT_s^r$ a surgical case s may depend on, C_{sj} resources $r \in R_j$ should be assigned during $[s_{sj}, s_{sj} + d_{sj})$.*

Constraint 6. *A resource can only be assigned to one surgical case at any given time and is used during the entire surgical phase it is assigned to (no pre-emption).*

Soft constraint 2. *Optional resource types RT_s^o for a surgical case s must be assigned as much as possible, i.e. shortages with respect to C_{sj} should be minimized.*

5.2.3 OR and resource considerations

Resource efficiency

Next to handling the dependency of surgical cases on resources, the aim is also to schedule resources assigned to surgical cases as efficiently as possible. Therefore some measures of resource efficiency are considered as well.

Firstly, resources considered in this paper are assumed to be *mobile*. This is a reasonable assumption given that stationary resources are fixed to a specific OR, and are thus scheduled implicitly together with the OR. Even though resources are assumed mobile, some should only be used in few OTs, and some should not be transferred too much (e.g. large equipment). Therefore, for such resources the aim is to minimize the number of ORs they are used in, and the transfers. However, other resources may not have such a restriction. A surgeon assigned to two ORs, may want to alternate surgical cases in different rooms to avoid non-surgical tasks that occur at the start/end of a surgery, such as anaesthesia and closing/cleaning. Thus, transfers for such resources (i.e. surgeons, but also others) should not be constrained/minimized, but the number of ORs these resources are used in should not exceed their assigned ORs.

To accommodate this, the following is defined:

- C_r : the number of ORs a resource r may be used in.
- $R^{TRAN} \subseteq R$: the set of resources for which transfers between ORs must be minimized.

These requirements are then formalized by:

Soft constraint 3. *The number of ORs a resource r is used in, should be smaller than C_r, i.e. the surplus should be minimized.*

Soft constraint 4. *For resource $r \in R^{TRAN}$, transfers between ORs should be minimized.*

Secondly, next to transfers between ORs it may also be important that resources are not left idle between surgical cases. For example, surgeons may prefer that their surgical cases are not scattered during the day, but are grouped together. Idle time between surgical cases may need to be minimized, for ensuring maximal efficient usage of a resource. Let $R^{IDLE} \subseteq R$ denote the set of resources for which the idle time between surgical cases must be minimized. Then:

Soft constraint 5. *The total idle time between surgical phases to which r is assigned should be minimized, for all $r \in R^{IDLE}$.*

Resource affinity

Meskens et al. [56] point out that generally some affinities may exist between surgical staff members, i.e. some people work better together than others. Their model considers a positive-valued affinity matrix defined between individual surgical staff members to address this aspect. Affinities are ranked on a range of 0 to 9, with 0 denoting incompatible and 9 denoting strong preference.

In this work, a simplified version of this idea is used that distinguishes between negative, neutral and positive affinities. Therefore, an affinity matrix $AF(r_1, r_2)$ with $r_1, r_2 \in R$, is defined as follows (assuming minimization of conflicts):

$$AF(r_1, r_2) = \begin{cases} 1 & \text{if } r_1, r_2 \text{ have a } \textit{negative} \text{ working affinitiy,} \\ -1 & \text{if } r_1, r_2 \text{ have a } \textit{positive} \text{ working affinity,} \\ 0 & \text{if } r_1, r_2 \text{ have a neutral working affinity.} \end{cases} \quad (5.1)$$

The aim is consider these affinities when assigning resources to surgical cases.

Soft constraint 6. *Positive working affinities should be maximized when assigning resources to surgical cases, whereas negative working affinities should be avoided. That is, the total affinity cost should be minimized.*

Note that no strong incompatibility is defined.

OR idle time

Maximal throughput and efficient occupation of ORs are also ensured, by minimizing the idle time between surgeries scheduled in an OR.

Soft constraint 7. *For every OR, the total idle time between surgeries should be minimized.*

Note that this goal does not leave time between surgeries for e.g. cleaning the OR. We assume that such ‘setup times’ are considered in the surgical case duration d_s. Surgical phases can be defined, accordingly, to end when cleaning should start.

Ordering constraints

For medical and practical reasons, there is a preferential ordering of surgical cases within any OR. For example, typically patients with latex allergies are operated on before other surgical cases, while patients who may be infectious are operated on after all other cases (to avoid contamination). Another common case is that children are operated earlier during the day, after which adults follow. This work assumes that such sequencing rules can be captured by an ordering p_s, specific to each case s, and:

> **Constraint 7.** *Within any OR, surgical cases should be performed in order of increasing priority p_s.*

5.2.4 Objective function

The main aim of the surgical case scheduling problem is scheduling as many surgical cases as possible (within availabilities). As a secondary objective, the number of OR days, i.e. days individual ORs are occupied, is minimized. Essentially the combination of these two objectives maximize efficient usage of the OT.

In addition, soft constraints 1-7 should also be considered. These three elements are combined in a weighted sum objective function:

$$\textbf{Minimize } W_S \cdot \text{unscheduled cases} \tag{5.2}$$

$$+W_R \cdot \text{OR days} \tag{5.3}$$

$$+\sum_{i=1}^{7} W_i \cdot \text{penalty soft constraint } i \tag{5.4}$$

with W_S, W_R, W_i denoting weights indicating relative importance of the objectives. This work sets weights such that $W_S \gg W_R \gg W_i$, as scheduling surgical cases is of primary importance, followed by minimizing the number of OR days.

5.3 Algorithmic approach

Previous research efforts have mostly focused on approaches based on mathematical programming formulations and related techniques (e.g. integer

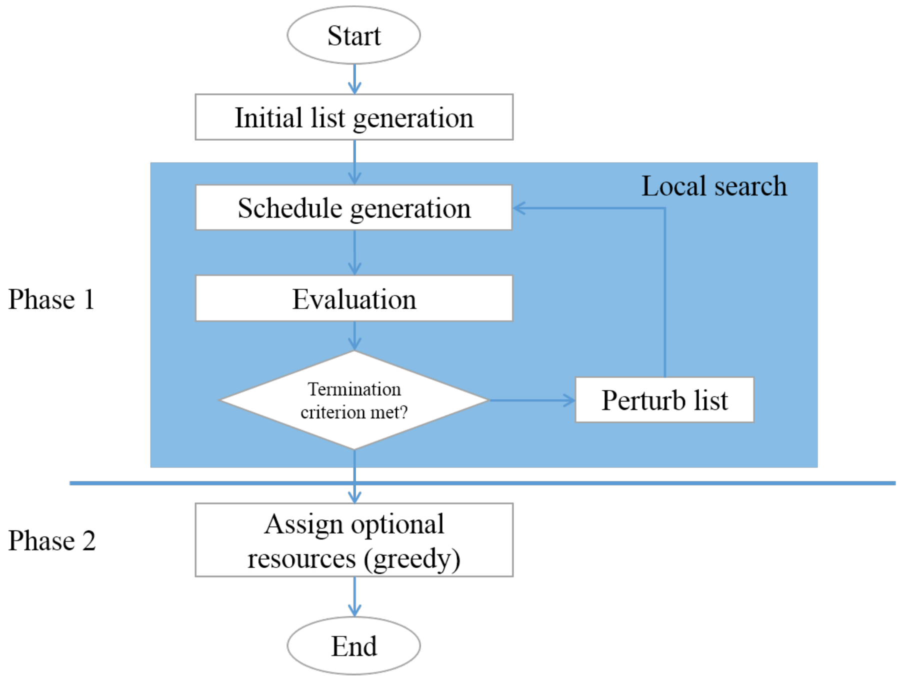


Figure 5.2: General overview of the heuristic approach.

programming [53], column generation [12]) and constraint programming [56, 82]. However, such approaches often suffer from the dimensionality of the problem, limiting scalability. This study has thus opted for a heuristic two phase approach in order to deal with the generalized problem definition, which may involve many resources.

A general overview of the two phase approach is presented in Figure 5.2. The approach is based on a list decoding procedure for generating feasible schedules. This procedure takes as input a list of surgical cases in a specific order and produces a schedule that adheres to all hard constraints of the problem. Next, a local search algorithm is used to manipulate this list of surgical cases in order to find new feasible schedules of better quality. Finally, in a second phase, optional resources are greedily assigned to surgical cases.

Notation	Description
(s, o, d, AR)	A tuple assigning s to room o on day d, with resource assignments AR.
$AR = \{(j, r) \| j \in RT^r_s, r \in R_j\}$	Resources r assigned to resource type j for case s.
$T_{o,d}, T_{r,d}$	OR schedule / resource schedule for OR o, resource r on day d.
$T(s)$	Starting time of case s.
$T_r(s)$	Starting time of resource r for case s.

Table 5.3: Summary of data structures and decision variables.

5.3.1 Solution representation and list decoding procedure

The list decoding procedure takes as input an ordered list of surgical cases in which each surgical case is already assigned to an OR o, a day d and to a set of resources AR. From this list, a feasible schedule/solution is constructed. Therefore, the main decision variable in this approach is an ordered list of tuples, each consisting of a surgical case s, an OR o, an assigned day d and a set of assigned required resource type/resource pairs $AR = \{(j, r)|j \in RT^r_s, r \in R_j\}$, or formally:

$$< (s_1, o_1, d_1, AR_1), (s_2, o_2, d_2, AR_2), \ldots, (s_N, o_N, d_N, AR_N) > \qquad (5.5)$$

Note that for resource assignments, only required resource types are considered. Optional resources are handled differently, see Section 5.3.4.

The ordering must be feasible with respect to the ordering/priority constraint (Constraint 7) for any day d and OR o, i.e. (s_k, o_k, d_k, AR_k) should appear earlier in the list than (s_l, o_l, d_l, AR_l) if $d_k = d_l$,$o_l = o_k$ and $p_{s_k} < p_{s_l}$ for any $1 \leq k < l \leq N$.

Given this priority feasible list, the list decoding procedure produces a feasible schedule as follows.

1. **Initialize data structures**: the main data structures of importance are representations of a schedule for both the ORs and the resources. These data structures will hold the partial schedules of all surgical cases/surgical phases assigned to an OR/resource.

   A schedule is implemented as an *interval tree*, a balanced binary tree data structure for storing intervals over the real numbers. Intervals are stored ordered first by decreasing start time, and second by decreasing end time

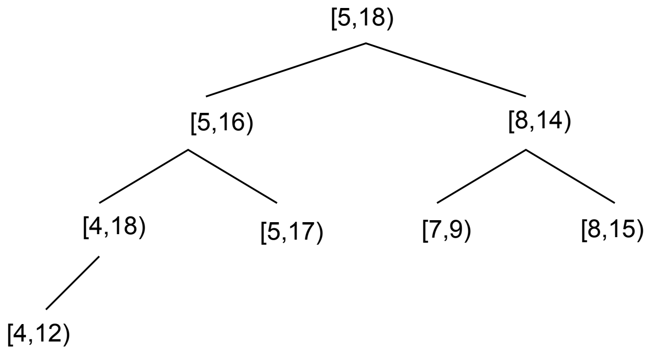


Figure 5.3: An example interval tree holding the intervals (in order) $[4, 12)$, $[4, 18)$,$[5, 16)$, $[5, 17)$,$[5, 18)$,$[7, 9)$,$[8, 14)$, and $[8, 15)$.

(when start times are equal). Figure 5.3 shows an example interval tree.

An interval tree is well suited for testing whether a point/interval is contained in/overlaps with an interval in the tree. Such operations can be performed in $O(\log n)$, if the tree is balanced. Finding the first interval before/after a query point or interval can also be found in $O(\log\ n)$. Finally, given a node in the tree, the next node in the ordering can be found in $O(\log n)$. Cormen et al. [18] provide a thorough introduction to interval trees, and how they can be implemented through augmenting a Red Black tree, a self balancing binary tree data structure.

For both ORs and resources, interval trees are used to store all relevant intervals (surgical case start + duration for ORs, surgical phase start + duration for resources), where each node also stores a pointer to the relevant surgical case. The schedules of OR $o \in O$ are denoted $T_{o,d}$ for each day of the planning horizon $d \in H$. The schedules of resource $r \in R$ are denoted $T_{r,d}$. Similarly, the availability intervals of an OR o/resource r are stored in interval trees $AV_{o,d}/AV_{r,d}$ for each day d.

2. **Variable definitions:** Let S' denote the set of all scheduled surgical cases up to this point. Let $S'_{o,d}$ denote the subset of S' that were scheduled in OR o, on day d. Let $S'_{r,d}$ denote the subset of S' that use resource r on day d. Let $T(s)$ denote the scheduled start time of surgical case s. Let $T_r(s)$ denote the scheduled start time of a surgical phase for assigned resource $r \in AR$.

3. **Schedule cases:** For each tuple (s, o, d, AR) in the priority feasible list:

(a) Find the earliest starting time t for case s as:

$$t = \max_{\substack{s' \in S'_{o,d}: \\ p_{s'} < p_s}} (T(s') + d_{s'})$$

i.e. the earliest starting time is after the end time of the last surgical case of a lower priority case in the same room on the same day.

(b) Check if $[t, t + d_s)$ is contained in an interval in $AV_{o,d}$. If not, find the next interval $[u, v)$ in $AV_{o,d}$ and $t := u$. If no such interval $[u, v)$ exists, there is no more availability of room o on day d. Leave s unscheduled and go to 3.

(c) Check if $[t, t+d_s)$ overlaps with any interval $[T(s'), T(s')+d_{s'}) \in T_{o,d}$. If yes, $t := T(s') + d_{s'}$ and go to 3b.

(d) Check if $[t + s_{sj}, t + s_{sj} + d_{sj})$ is contained in an interval in $AV_{r,d}$, for each $(j, r) \in AR$. If no resource r is available, find the next interval $[u, v)$ in $AV_{r,d}$ and $t := u - s_{sj}$. If no such interval $[u, v)$ exists, there is no more availability for resource r on day d. Leave s unscheduled and go to 3.

(e) Check if $[t+s_{sj}, t+s_{sj}+d_{sj})$ overlaps with any interval $[T_r(s'), T_r(s')+d_{s'j}) \in T_{r,d}$ for any $(j, r) \in AR$. If yes, $t := T(s') + d_{s'j} - s_{sj}$ and goto 3b.

(f) Schedule surgical case: $S' := S' \cup \{s\}$, $T(s) := t$, Insert$([T(s), T(s) + d_s), T_{o,d})$, and Insert$([T(s) + s_{js}, T(s) + s_{js} + d_{js}), T_{r,d})$ for each $(j, r) \in AR$.

At the end of this procedure, the result is a feasible schedule for all $s \in S'$, while $s \in S \backslash S'$ are left unscheduled.

Essentially, the algorithm just described constructs a schedule in the order defined by the priority feasible list. At each step, the algorithm maintains a schedule for each OR/resource that holds when it is available or occupied. The main loop defined in step 3 scans these schedules in order to find the earliest time at which the current surgical case s can be inserted, respecting its assigned OT and resources. If no such insertion position is found on day d, due to lack of available time, the surgical case s is left unscheduled.

5.3.2 Schedule evaluation

Given a priority feasible list l, its corresponding schedule can be computed using the list decoding procedure described in Section 5.3.1 (denoted

GenerateSchedule(l)):

$$(T(s), S', T_{o,d}, T_{r,d}, S'_{o,d}, S'_{r,d}) := \text{GenerateSchedule}(l) \tag{5.6}$$

with $T(s)$ containing the scheduled start time of each case s, S' the set of scheduled cases, $S'_{o,d}, S'_{r,d}$ the set of scheduled cases for OR o, resource r on day d and $T_{o,d}, T_{r,d}$ containing the occupied intervals of each OR o/resource r.

The schedule can be evaluated on the following objectives:

- Number of surgical cases left unscheduled $= |S \backslash S'|$
- Number of OR days $= \#OR$ with:

$$\#OT = \sum_{d \in H} \sum_{o \in O} (T_{o,d} \neq \emptyset) \tag{5.7}$$

  where $(T_{o,d} \neq \emptyset) = \begin{cases} 1 & \text{If } |T_{o,d}| > 0 \\ 0 & \text{otherwise.} \end{cases}$

- Number of surgical cases scheduled in preferred room $= |Pref|$ with

$$Pref = \{(s, o, d, AR) \in l | o \in O_s^1 \text{ and } s \in S'\} \tag{5.8}$$

- Number of surgical cases scheduled in 'if-necessary' room $= |IfNec|$ with

$$IfNec = \{(s, o, d, AR) \in l | o \in O_s^3 \text{ and } s \in S'\} \tag{5.9}$$

- Number of resource 'overloads':

$$Overload = \sum_{d \in H} \sum_{r \in R} \max\left(|O_{r,d}| - C_r, 0\right) \tag{5.10}$$

  with:

$$O_{r,d} = \{o \in O | \exists (s, o, d, AR) \in l \text{ and } s \in S'_{r,d}\} \tag{5.11}$$

  denoting the distinct ORs in which a resource r is scheduled on day d.

- Number of resource transfers for resource r on day d: let $l^{S'}_{r',d'}$ denote the *sub*list of l containing (s, o, d, AR) for which $s \in S'$ (it is scheduled), $d = d'$ (assigned to day d') and $\exists (j, r') \in AR$ (has r' assigned to it).

  Then the number of transfers for resource r on day d can be determined by considering all adjacent pairs (s_k, o_k, d_k, AR_k) ,(s_l, o_l, d_l, AR_l) in $l^{S'}_{r,d}$, and checking if $o_k \neq o_l$.

- Total OR idle time:

$$TotalORIdleTime = \sum_{d \in H} \sum_{o \in O} \left(Max(T_{o,d}) - Min(T_{o,d}) - \sum_{s \in S'_{o,d}} d_s \right) \tag{5.12}$$

  with $Min(T_{o,d})$,$Max(T_{o,d})$ denoting the first start (last end) time of a surgical case in room o on day d.

- Total resource idle time:

$$TotalResourceIdleTime = \sum_{d \in H} \sum_{r \in R^{IDLE}} \left(Max(T_{r,d}) - Min(T_{r,d}) - \sum_{(T(s)+s_{sj}, T(s)+s_{sj}+d_{sj}) \in T_{r,d}} d_{sj} \right) \tag{5.13}$$

  with $Min(T_{o,d})$,$Max(T_{o,d})$ denoting the first start (last end) time of a surgical case in room o on day d.

- Total resource affinity:

$$TotalResourceAffinity = \sum_{s \in S'} \sum_{(j_1, r_1),(j_2, r_2) \in AR} AF(r_1, r_2) \tag{5.14}$$

5.3.3 Local search procedure

The schedule constructed by the list decoding procedure may be arbitrarily bad. Complex resource dependencies may not be resolved due to the sequence of the cases in the priority feasible list, or a particular resource may be assigned to too many surgical cases. Therefore, the list decoding procedure is used within a local search framework. It enables modifying both the sequence of the surgical cases in the priority feasible list, as well as the assigned resources, OR or operating day of the surgical cases.

The following local search operators have been developed:

- Shift (**S**): given a priority feasible list l, shift one tuple (s, o, d, AR) to a new position, maintaining the priority feasible nature of the list.
- Change Day (**CD**): given a tuple $(s, o, d, AR) \in l$, replace d by $d' \in D_s$.
- Change OR (**COR**): given a tuple $(s, o, d, AR) \in l$, replace o by $o' \in O_s$.

- Change Assigned Resources (**CAR**): given a tuple $(s, o, d, AR) \in l$, with $AR = \{(j, r) | j \in RT_s^r, r \in R_j\}$, select a required resource (j, r) and a resource $r' \in R_j$ and replace (j, r) by (j, r').

A stochastic improving-or-equal local search algorithm has been developed applying these operators to improve an initial feasible priority list and corresponding schedule. Pseudocode of this procedure is presented in Algorithm 6.

Algorithm 6 Local search procedure

```
Require: f : l ↦ ℝ                               ▷ f applies list decoding and evaluation to l
  l ← RandomPriorityFeasibleList()              ▷ Rand. assigns o, d, AR from
  O_s, D_s, RT_s^r × R
  i ← 0
  while termination criterion not met do
      N ← SelectNeighbourhood(S,CD,COR,CAR)
      l' ← N(l)                                 ▷ Sample a neighbouring solution from l
      if f(l') ≤ f(l) then                      ▷ Only accept improving/equal solutions
          l ← l'
      end if
      i ← i + 1
  end while
  return l
```

5.3.4 Optional resource assignment

After the local search phase has finished, the final schedule is constructed. Up to this point, only required resource dependencies have been considered in the approach, while optional resource dependencies have been left unassigned. Essentially, optional resource dependencies could have been handled in a similar fashion as required dependencies. Optional resources can be assigned to a surgical case s, be scheduled by the list decoding procedure, and be manipulated by the local search. For this, the list decoding procedure should be suitably modified to not prohibit a case from being scheduled due to unavailability of an optional resource.

However, a study on real-life problem instances (presented in Section 5.5.1) revealed that optional resources were quite numerous. Treating the optional resources in the same manner as required resources would therefore slow down the approach unacceptably.

A two-phase approach has been developed to cope with this problem, performing the optional resource assignment after the surgical case schedule has been constructed. The optional resource assignment is implemented using a greedy approach: it considers the scheduled surgical cases one by one, and assigns the resource that minimally increases the objective function.

5.4 Modelling examples

To illustrate the flexibility of the resource requirements, we present two common practical considerations in surgical case scheduling and show how they can be modelled using generalized resource dependencies.

5.4.1 Post-anaesthetic care unit

Availability of a bed in the post-anaesthetic care unit (PACU) is an important consideration for scheduling surgical cases. Generally, patients undergo surgery under local, regional or general anaesthesia, and require post-operative monitoring (typically a few hours) in the PACU to assess recovery thereof. A bed in the PACU must be available when surgery ends to ensure proper monitoring. If no beds are available for some time, surgeries may be delayed or even postponed to a later date.

The following shows how such a dependency can be modelled using the generalized resource constraints.

- Define a resource type j_{PACU} denoting the PACU. Define as many resources r as there are beds in the PACU, with $RT_r := \{j_{PACU}\}$.
- For each surgical case s (assuming all surgeries require post-anaesthetic care), add a required resource dependency on j_{PACU}, i.e. $RT_s^r := RT_s^r \cup \{j_{PACU}\}$ with $C_{sj_{PACU}} = 1$ and $s_{sj_{PACU}} = d_s$, $d_{sj_{PACU}} =$ Required monitoring time in the PACU. Thus, a required resource dependency is defined with surgical phase starting *after* the surgical case *ends*. Therefore, the surgical case can only be scheduled at a time when bed availability in the PACU can be ensured after the surgery ends.

5.4.2 Instrument kits

Another common practical consideration is the usage of *instrument kits*, standard sets of surgical tools (e.g. scalpels, scissors, clamps). Clearly such tools are necessary during surgery and thus need to be available. It is important to account for the fact that these also must be cleaned and sterilized after each surgery. This cleaning process is typically performed in a dedicated facility, which may take significant time. The *turnaround* time of this cleaning process can therefore not be neglected. When instrument kits are in limited supply (some kits are specialized for example), this may require extra attention during planning.

Again such a dependency can be modelled

- Define resource types $j_{instr1}, j_{instr2}, \ldots, j_{instrK}$ as much as there are different kinds of instrument kits (e.g. K). Define as many resources r as there are instrument kits of each type, with $RT_r := \{j_{instr1}, j_{instr2}, \ldots\}$. (Large kits may serve multiple purposes thus it is possible that $|RT_r| > 1$).
- Add resource dependencies to surgical cases s, accommodating their instrument kit requirements, e.g. $RT_s^r := RT_s^r \cup \{j_{instr1}\}$ and accordingly s_{sj}, d_{sj}, with d_{sj} sufficiently long to account for cleaning turnaround time.

5.5 Computational experiments

5.5.1 Experimental setup

Data for the surgical case scheduling problem was provided in the context of the joint research project between the research group and Dotnext. A dataset of 52 problem instances was obtained from a hospital managing an OT consisting of 24 ORs, of which 18 are general purpose ORs and 6 are specialized. General characteristics of this data can be found in Table 5.4.

We distinguish the problem instances by the planning horizon, which ranges from 1 to 7 days. The planning horizon relates to the problem size. Single-day instances have fewer appointments to be planned than multi-day instances. As can be seen in Table 5.4, the number of resources under consideration is quite high. In general, the surgical cases have only one required resource type, a surgeon, and 2-3 optional resource requirements, representing the remaining surgical team (anaesthesiologists, nursing staff). The specified surgical phase $[s_{sj}, d_{sj})$ for surgeons is smaller than the surgical case interval $[0, d_s)$ (i.e. surgeons only need to be present during the incision phase). In addition, surgical cases were always fixed to a specific surgeon. Incorporation in the presented model can be achieved by specifying a unique 'resource type' specific to each surgeon. No MSS is imposed, therefore all ORs are available for scheduling

$\|H\|$	# instances	Avg. $\|R\|$	Avg. $\|S\|$
1	11	276.1	86.5
2	9	285.2	175.2
3	7	288.6	258.4
4	11	288.8	247.9
5	8	292.1	306.5
6	4	293.5	339.5
7	2	293.5	429.5

Table 5.4: General problem instance characteristics. Instances are grouped by their planning horizon, ranging from 1 to 7 days.

Weight	Value
Planned appointments	1 000 000
# OR days	10 000
Resource overload	10
Unwanted transfers	1
Optional resources left unplanned	100
Possible room assignment	1
If-necessary room assignment	100
Affinity	1
OR idle time	0.1
Resource Idle time	0.01

Table 5.5: Weights determining relative importance of different objectives and penalties.

surgical cases. Finally, weights for the instances were determined in preliminary testing to reflect the relative importance of the different criteria. These are reported in Table 5.5.

The algorithm has been coded in Java 1.7. All tests have been performed on a workstation computer equipped with two eight-core Intel Xeon 2650 v2 2.6 GHz processors and 128 GB of main memory (RAM), running a Linux-based operating system. Only one processing thread is used per test (the algorithm does not employ parallelism), and therefore this system was used to perform up to 16 tests in parallel (limiting available memory to 8 GB for each test).

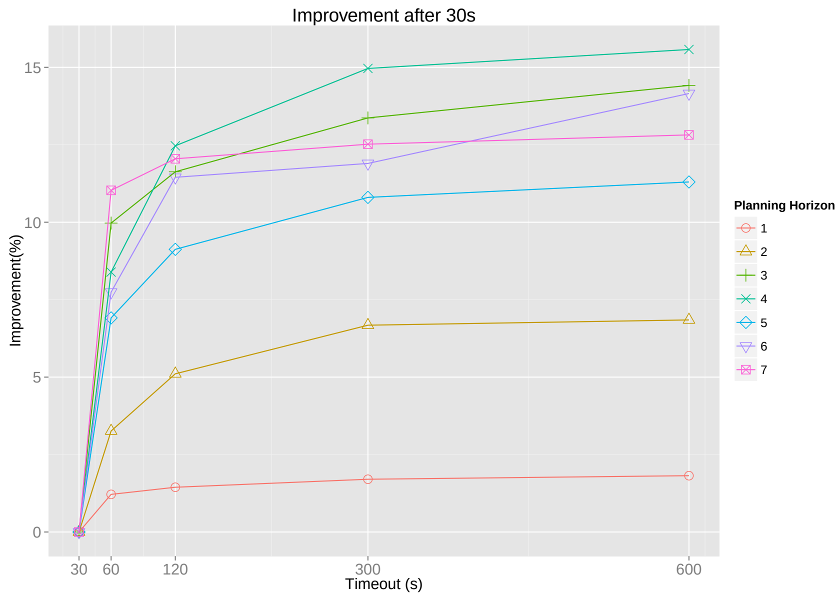


Figure 5.4: Relative improvement of the local search algorithm after 30 seconds, with respect to the size of the planning horizon.

5.5.2 Results and discussion

Convergence of the algorithm

One important aspect is whether or not the algorithm is able to converge in a reasonable time, considering that the number of resources (and thus the problem size) is quite large. Figure 5.4 shows the relative improvement that can still be found after an initial 30 seconds of running the local search algorithm. It is clear that for small instances (1-2 day instances) the algorithm is able to converge in a relatively short time span of 1-2 minutes. The objective of the presented research was to construct schedules for day instances in a reasonable time and to allow for near-interactive use. For the larger planning horizons (up to 1 week), it is clear that additional time is required (up to 10 minutes) for the improvement process to start to converge. However, in a weekly planning setting the required time is less of an issue.

Comparison with practice

Another objective in surgical case scheduling is the quality of the schedules. The generated schedules were compared with schedules made by manual planners. However, it must be noted that these schedules rarely were completely *feasible*: overlap could be detected between surgical cases, surgical phases would overlap for some resources, or surgical cases were planned outside of availabilities of the ORs/resources. For the comparison, these surgical cases were considered unscheduled. In the case of overlap between two surgical cases (or their required resources), only one is left unscheduled. Infeasibilities due to overlap for optional resource requirements were handled by leaving the optional resource unassigned.

Table 5.6 reports on the results obtained by the heuristic approach, using different timeout values T (30, 60, 120, 300 and 600 seconds) and the manually constructed schedules (last column).

Clearly, the presented approach is able to schedule more surgical cases without violating any of the hard constraints. In addition, it is able to do this in less OT days, although this may be related to an implicit MSS being considered by human planners. With respect to the secondary soft constraints 1-7, the approach is able to plan more optional resources (feasibly) and minimize resource affinities. Evidently, this results in higher total resource overload and more resource transfers, as more resources are assigned. However, it is assumed that assigning optional resources takes precedence over their performance. Finally, operating room idle time is reduced to a negligible level and resource idle time is reduced, showing that the approach is able to produce tighter schedules.

	Our approach with timeout T (s)					Human
Performance measure	$T = 30$	$T = 60$	$T = 120$	$T = 300$	$T = 600$	planner
# Appointments left unplanned	**13.4**	**13.2**	**13.1**	**13.0**	**13.0**	23.2
# OR days	**53.4**	**50.9**	**49.1**	**47.7**	**47.1**	62.0
Total room idle time (minutes)	**140.8**	**73.8**	**42.0**	**23.6**	**19.2**	2163.0
Total resource idle time (minutes)	**4860.7**	**4493.7**	**4287.5**	**4141.2**	**4083.1**	6749.3
# Optional resources left unplanned	**186.4**	**186.2**	**186.1**	**185.8**	**185.7**	376.4
Total resource overload	52.5	39.2	30.5	24.6	22.6	4.8
# Total resource transfers	68.5	55.1	45.2	37.7	35.1	8.0
Resource affinity cost	**-6.4**	**-7.1**	**-7.8**	**-8.3**	**-8.6**	0.0
# Appointments not in *"preferred"* room*	0.0	0.0	0.0	0.0	0.0	0.0
# Appointments in *"if-necessary"* room*	0.0	0.0	0.0	0.0	0.0	0.0

Table 5.6: Results of the heuristic approach on the different measures that are considered. Performance is reported for different time-outs: 30, 60, 120, 300 and 600 seconds. A comparison to the performance measures on schedules made by human planners is also provided (last column). Bold highlighted results indicates our approach obtained a better result than the human planner.
*For each surgical case s, all ORs were specified as possible. I.e. no preference / if-necessary rooms were specified.

5.6 Conclusion

A decision support model for multi-day OT scheduling has been presented that encompasses many considerations from practice. The goal has been to develop a sufficiently general and flexible model to cover different practices to OT scheduling in hospitals. To this end, generalized resource dependencies were introduced to cope with a broad variety of scheduling considerations: human dependencies (surgeons, anaesthesiologists, instrumentalists, nurses) as well as material dependencies (e.g. large surgical equipment), dependencies during specific surgical phases, etc. The aim of the model is to assist in scheduling as many surgical cases as possible, within availabilities of all considered ORs and resources, to reduce the number of ORs that need to be opened for performing surgical cases, to minimize violations of soft constraints and to optimize several measures of resource efficiency.

A heuristic approach has been developed to address this feature-rich problem. The algorithm scales favourably with problem size. In addition, the algorithmic complexity of the schedule generation approach does not depend on the precision of scheduling and is therefore able to schedule up to any precision of time. The approach has been validated on a set of problem instances obtained from a hospital. Computational experiments on these problem instances show that the approach is able to schedule more surgical cases in a feasible way, whilst further decreasing the required number of ORs. In addition, secondary resource performance and efficiency measures are also improved.

Chapter 6

Conclusion

6.1 Summary and contributions

The present dissertation has focused on the development of computational models and algorithms for providing decision support for the admission process at the operational decision level. Three planning and scheduling processes that occur during the admission process have been studied: the assignment of patients to hospital wards and rooms, the admission process for elective surgical patients and the daily/weekly scheduling process for the operating theatre.

Patient-to-room assignment planning: The patient-to-room assignment planning process has been studied in two chapters, reporting on the application of decision support models in a dynamic setting and the theoretical complexity of the underlying assignment problem in which gender separation is considered.

In Chapter 2, an existing problem definition of the patient-to-room assignment problem (PA) has been extended from a static, offline setting to a dynamic, online one. The main research question was to determine whether or not decision support models can support and improve decision making in an online setting, where uncertainty on patient admissions and patients' length of stay (LOS) may deteriorate decision making. It is shown that an anticipative approach, considering lookahead on registered future admissions, may improve over a reactive policy, even when presented with inaccurate estimates of patients' LOS and unforeseen admissions from the emergency department. Even when hospital occupancy is high, and the flexibility for room planning is reduced, the second model is advantageous over the first. In addition, it is shown that allowing the flexibility of patient transfers does not necessarily improve decision making.

This work implies that the patient-to-room assignment process may be improved and better patient care and comfort may be achieved. The presented models can be applied in a central, hospital-wide admission system; supporting bed managers and admission officers in their daily task of assigning patients to hospital rooms. The models can be used each day to plan the room assignments for the patient arrivals of the upcoming day, and to reschedule room assignments on the next day, considering unplanned, urgent/emergency arrivals.

In Chapter 3, the main research question was to establish how the gender separation policy impacts the computational complexity of the PA problem. To this end, the Red-Blue transportation problem (Red-Blue TP) was defined as an abstraction of the PA problem. The Red-Blue TP is a new generalization of the well known transportation problem (TP) that considers a partitioning of the supply nodes and which requires that no two supply nodes from the different partitions may send supply to the same demand node. Being a generalization of the TP, the Red-Blue TP presents an interesting opportunity to study the effect of this type of *exclusionary constraints* (i.e. a separation policy between two groups), since it is well known that the TP can be solved using a polynomial time algorithm. The Red-Blue TP is shown to be an NP-Hard problem and this result stands in two special cases. This complexity result extends beyond the PA problem setting, as the Red-Blue TP may be identified as a special case of other practical problems, thus establishing their complexity.
Furthermore, two integer programming formulations for the problem are presented for which it is shown that one is stronger than the other. A computational study points out however, that this stronger formulation is not necessarily the best choice, as the overhead by a.o. additional variables makes the model computationally less efficient to solve.
A maximization variant of the Red-Blue TP is also presented, for which three algorithms are developed; two of which guarantee an approximation ratio of $\frac{1}{2}$. Furthermore, several variants of this 'Max-Red-Blue TP' are considered and the applicability of the approximation algorithms to these variants is discussed. Although the approximation algorithms are $\frac{1}{2}$-approximations in the worst case, the practical performance as validated on a set of instances with varying characteristics is much more distinct.

Admission scheduling: In Chapter 4, a robust admission scheduling approach is presented for scheduling elective surgical patients. The main research motivation was to make the admission scheduling for surgical patients less myopic. Whereas most existing literature focuses solely on the operating theatre performance, the presented approach also considers surgical ward capacity. In addition, the approach is to be used for decision making in an online, uncertain setting, with uncertainty on patients' length of stay and surgical durations. To this end, a chance-constrained stochastic admission scheduling model is presented that aims to minimize expected operating theatre costs and patient waiting time, and that avoids the risk of bed shortages at a fixed confidence level. A sample average approximation of the model is solved by a meta-heuristic algorithm and serves as the basis for four admission scheduling variants. In a computational study, it is shown that a stochastic approach may accurately consider variance in operating theatre costs and surgical ward usage. When the admission strategies are sufficiently flexible, this may lead to improved hospital performance in terms of either bed shortages or operating theatre costs, though at the cost of patient friendliness and patient waiting time. Decision makers will therefore closely need to evaluate to what extent the different performance measures impact hospital operations in order to find an acceptable balance.

Operating theatre scheduling: Chapter 5 discussed the operating theatre scheduling problem of determining both start times and dates for surgical cases over a fixed scheduling horizon. The main research question was how different resource dependencies (surgeons, anaesthesiologists, nursing and instrumenting staff, but also material requirements) can be taken into account during this process in a general and flexible way. To this end, a generalized resource dependency model is proposed along with several other considerations applicable to current operating theatre practice. A heuristic approach is presented that scales favourably with problem size, and that is validated on a set of test instances from practice. An important contribution is that the approach has been developed with the mindset to be implemented in a commercial software application of a software partner.

The literature and the state of the art on operating theatre scheduling techniques has matured considerably in the past decade, as can be seen in several literature reviews [13, 34]. Practical case studies in pilot hospitals have shown that surgical scheduling techniques can effectively increase performance. Although larger hospitals may have the research staff and the IT resources to pick up and implement these techniques in their own (often in-house developed) hospital information systems, smaller and mid-size hospitals rely on commercial software available from different vendors. However, from informal discussions with hospitals and software vendors we have observed that very few of these techniques are finding their way to implementations in commercial software. Even worse, they may not even be aware that these algorithmic techniques exist, as they typically do not have access to the scientific repositories and journals where these findings are published. Discussions with software developers, and implementation of the research results in their applications, may be an important step for bringing the research results on operating theatre scheduling techniques into commercial software applications. If the ability to automatically plan surgical cases becomes a crucial selling point, others vendors will have no other option to also explore these techniques in order to remain competitive.

6.2 Perspectives for future work

Clearly, the application possibilities of operations research to health care services are broad. Even the restricted scope that was considered in this dissertation leaves many open ends and interesting directions for future research. Therefore, to conclude this dissertation, some of the author's ideas are presented next.

Patient-to-room assignment planning: One interesting direction for future research is to look into further integration of the scheduling process of patients and the room assignment process. Currently, the PA problem takes the scheduled arrival times of elective patients as input, as found by a scheduling process such as the one presented in Chapter 4. It is therefore limited in what can be achieved. If the two processes could be integrated, or if the scheduling process could at least take into account some information from the room assignment process, better global results may be obtained. This is, however, a difficult research question as the two processes have conflicting objectives: scheduling processes typically aim for high utilization of resources such as beds and the operating theatre, whereas the difficulties that the room assignment process tries to solve follow from a high occupancy of the wards. In addition, the problem dimension increases significantly since both a temporal and a spatial decision need to be taken. Different solution approaches may be more appropriate. One possible approach would be to apply some form of decomposition with feedback between the two sub-problems, such as for example Logic-based Benders' decomposition [39]. In such an approach, the scheduling problem would take the role of the master problem which determines admission dates for elective patients. The secondary/slave problem would be a PA problem which can determine a lower bound on the patient-to-room assignment cost and gender costs for the given admission schedule. These costs can then serve as feedback to the master problem to find a new admission schedule that may produce a better global solution.

Admission scheduling: One open end that has not yet been considered, is to take into account emergency admissions. Clearly, emergency admissions must be considered in order to leave sufficient remaining capacity in surgical wards and in the operating theatre. Essentially, this should not be a problem in the current approach: additional random variables can be modelled to represent surgical ward usage and operating theatre usage by emergency patients for each day of the scheduling horizon. In addition, a method must be available for sampling these random variables. Two approaches can be employed. A forecasting method can be implemented to estimate the arrival rate of emergency patients, that can then be used to simulate emergency patient arrivals. Combined with fitted distributions of length of stay and surgery duration (specifically fitted to emergency patients), samples can be generated for the surgical ward usage and the operating time usage. Another approach could be to fit distributions to the measured bed usage and operating time usage by emergency patients on historical data, for each day of the week/month/year/etc. These distributions can then be sampled directly.

Operating theatre scheduling: The model that has been presented in this dissertation is rather general, but still not complete. One particular element related to resources that has not yet been implemented are non-renewable resources, or consumables. This should be rather straightforward to implement in the current resource model. Another aspect that has not been considered is the inherent uncertainty related to surgical durations. Variance on surgical durations may be taken into account in the schedule generation procedure. Instead of scheduling a new surgical case at the earliest possible insertion point, the procedure should introduce sufficient slack with respect to earlier surgical cases, in both the operating room schedules and in the resource schedules.

Appendix A

Red Blue TP instance generation procedure

A.1 Instance generation procedure

Algorithm 7 generates Red-Blue TP instances according to the following parameters:

- $|S|, |D|$: the number of supply and demand nodes,
- PR: the percentage of red supply nodes in the graph,
- DEN: the density of the graph,
- $S_{\max}$: the maximum supply for any given supply node,
- $C_{\max}$: the maximum cost (or profit, for Max-Red-Blue TP) for any given edge in the bipartite graph,
- *seed*: a seed value for the pseudo-random number generator.

Supply nodes are generated with supply a_i randomly selected in $[1, S_{\max}]$. The procedure ensures that total supply meets total demand by randomly dividing total supply over $|D|$ demand nodes (see Figure A.1). First, $|D| - 1$ unique numbers between 0 and S_{total} (the total supply) are generated. Next, these numbers are sorted and the demand nodes are then generated as the pairwise difference between these numbers. Finally, to reduce the density of the graph,

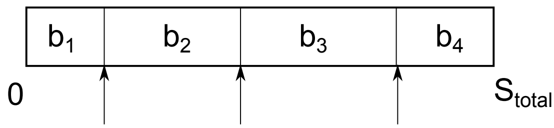


Figure A.1: Procedure for randomly generating demand, matching supply. The procedure generates $|D|-1$ random, unique numbers strictly between 0 and S_{total}. The numbers then divide the total supply over $|D|$ demand nodes.

edges of the graph are randomly selected and removed (indicated by setting $c_{ij}=-1$). 7.

Algorithm 7 Red-Blue TP instance generation procedure

Require: $|S|, |D|, PR, DEN, S_{\max}, C_{\max}, seed$

RAND $\leftarrow$ *seed*
$R \leftarrow \{1, \dots, PR \cdot |S|\}, B \leftarrow \{PR \cdot |S| + 1, \dots, |S|\}$
$S \leftarrow R \cup B,\ D \leftarrow \{1, \dots, |D|\}$
$S_{\text{total}} \leftarrow 0$
for $i \leftarrow 1, \dots, |S|$ **do**
    $a_i \leftarrow \text{RAND}(1, S_{\max})$
    $S_{\text{total}} \leftarrow S_{\text{total}} + a_i$
end for
$T \leftarrow \text{UNIQUERAND}(1, S_{\text{total}} - 1, |D| - 1)$
$SORT(T)$
$b_1 = t_1 - 0, b_2 = t_2 - t_1, \dots, b_j = t_j - t_{j-1}, \dots, b_{|D|} = S_{\text{total}} - t_{|D|-1}$
for $(i, j) \in S \times D$ **do**
    $c_{ij} \leftarrow \text{RAND}(0, C_{\max})$
end for
for $k \in (1 - DEN) \cdot |S| \times |D|$ **do**
    $(i, j) \leftarrow \text{UNIQUERAND}(S \times D)$
    $c_{ij} \leftarrow -1$
end forreturn Red-Blue TP$(S, D, (a_i), (b_j), (c_{ij}))$

Bibliography

[1] A. Agnetis, A. Coppi, M. Corsini, G. Dellino, C. Meloni, and M. Pranzo. Long term evaluation of operating theater planning policies. *Operations Research for Health Care*, 1(4):95–104, Dec. 2012. doi: 10.1016/j.orhc.2012.10.001.

[2] R. B. Bachouch, A. Guinet, and S. Hajri-Gabouj. An integer linear model for hospital bed planning. *International Journal of Production Economics*, 140(2):833–843, Dec. 2012. doi: 10.1016/j.ijpe.2012.07.023.

[3] G. Bai, J. Mao, and G. Lu. Grey transportation problem. *Kybernetes*, 33 (2):219–224, 2004. doi: 10.1108/03684920410514148.

[4] R. Bekker and P. M. Koeleman. Scheduling admissions and reducing variability in bed demand. *Health care management science*, 14(3):237–49, Sept. 2011. doi: 10.1007/s10729-011-9163-x.

[5] J. Beliën and E. Demeulemeester. Building cyclic master surgery schedules with leveled resulting bed occupancy. *European Journal of Operational Research*, 176(2):1185–1204, Jan. 2007. doi: 10.1016/j.ejor.2005.06.063.

[6] J. Beliën and E. Demeulemeester. A branch-and-price approach for integrating nurse and surgery scheduling. *European Journal of Operational Research*, 189(3):652–668, Sept. 2008. doi: 10.1016/j.ejor.2006.10.060.

[7] B. Bilgin, P. Demeester, M. Misir, W. Vancroonenburg, and G. Vanden Berghe. One hyper-heuristic approach to two timetabling problems in health care. *Journal of Heuristics*, 18(3):401–434, June 2012. doi: 10.1007/s10732-011-9192-0.

[8] M. E. Bruni, P. Beraldi, and D. Conforti. A stochastic programming approach for operating theatre scheduling under uncertainty. *IMA Journal of Management Mathematics*, Jan. 2014. doi: 10.1093/imaman/dpt027. Available online.

[9] E. K. Burke and Y. Bykov. A late acceptance strategy in hill-climbing for exam timetabling problems. In *PATAT 2008 Conference*, Montreal, Canada, 2008.

[10] B. Cao. Transportation problem with nonlinear side constraints a branch and bound approach. *Mathematical Methods of Operations Research (ZOR)*, 36(2):185–197, 1992. doi: 10.1007/BF01417216.

[11] B. Cao and G. Uebe. Solving transportation problems with nonlinear side constraints with tabu search. *Computers & Operations Research*, 22(6): 593–603, 1995. doi: 10.1016/0305-0548(94)00055-D.

[12] 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, Sept. 2009. doi: 10.1016/j.cor.2008.11.012.

[13] 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. doi: 10.1016/j.ejor.2009.04.011.

[14] B. Cardoen, E. Demeulemeester, and J. Van der Hoeven. On the use of planning models in the operating theatre: results of a survey in Flanders. *The International journal of health planning and management*, 25(4):400–14, 2010. doi: 10.1002/hpm.1027.

[15] S. Ceschia and A. Schaerf. Local search and lower bounds for the patient admission scheduling problem. *Computers & Operations Research*, 38(10): 1452–1463, Oct. 2011.

[16] S. Ceschia and A. Schaerf. Modeling and solving the dynamic patient admission scheduling problem under uncertainty. *Artificial Intelligence in Medicine*, 56(3):199–205, 2012. doi: 10.1016/j.artmed.2012.09.001.

[17] A. Charnes, W. W. Cooper, and G. H. Symonds. Cost horizons and certainty equivalents: an approach to stochastic programming of heating oil. *Management Science*, 4(3):235–263, 1958.

[18] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*, chapter Interval trees, pages 348–355. MIT Press and McGraw-Hill, 3rd edition, 2009. ISBN 0-262-03384-4.

[19] A. M. de Bruin, R. Bekker, L. van Zanten, and G. M. Koole. Dimensioning hospital wards using the Erlang loss model. *Annals of Operations Research*, 178(1):23–43, Oct. 2010. doi: 10.1007/s10479-009-0647-8.

[20] P. Demeester. Patient admission scheduling problem website, 2012. URL `http://allserv.kahosl.be/~peter/pas/`. Last accessed on August 4th, 2014.

[21] P. Demeester, W. Souffriau, P. De Causmaecker, and G. Vanden Berghe. A hybrid tabu search algorithm for automatically assigning patients to beds. *Artificial Intelligence in Medicine*, 48(1):61–70, 2010.

[22] B. Denton, J. Viapiano, and A. Vogl. Optimization of surgery sequencing and scheduling decisions under uncertainty. *Health Care Management Science*, 10(1):13–24, Nov. 2006. doi: 10.1007/s10729-006-9005-4.

[23] F. Dunke. *Online Optimization with Lookahead*. PhD thesis, Karlsruhe Institute of Technology, 2014.

[24] D. A. Etzioni, J. H. Liu, and C. Y. Ko. The aging population and its impact on the surgery workforce. *Annals of Surgery*, 238(2):170–177, 2003.

[25] M. Fackrell. Modelling healthcare systems with phase-type distributions. *Health Care Management Science*, 12(1):11–26, June 2008. doi: 10.1007/s10729-008-9070-y.

[26] H. Fei, C. Chu, and N. Meskens. Solving a tactical operating room planning problem by a column-generation-based heuristic procedure with four criteria. *Annals of Operations Research*, 166(1):91–108, Aug. 2008. doi: 10.1007/s10479-008-0413-3.

[27] R. O. Ferguson and G. B. Dantzig. The problem of routing aircraft. *Aeronautical Engineering Review*, 14:51–55, 1955.

[28] A. Fügener, E. W. Hans, R. Kolisch, N. Kortbeek, and P. T. Vanberkel. Master surgery scheduling with consideration of multiple downstream units. *European Journal of Operational Research*, 239(1):227–236, Nov. 2014. doi: 10.1016/j.ejor.2014.05.009.

[29] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. WH Freeman & Co., 1979.

[30] D. Gartner and R. Kolisch. Scheduling the hospital-wide flow of elective patients. *European Journal of Operational Research*, 233(3):689–699, Sept. 2014. doi: 10.1016/j.ejor.2013.08.026.

[31] C. Gogos, C. Valouxis, P. Alefragis, and E. Housos. Attacking google's machine reassignment problem using mathematical programming, metaheuristics and heuristics. In *25th European Conference on Operational Research (EURO2012)*, page 158, Vilnius, Lithuania, July 2012. Abstract.

[32] D. Goossens and F. C. R. Spieksma. The transportation problem with exclusionary side constraints. *4OR, A Quarterly Journal of Operations Research*, 7(1):51–60, 2009.

[33] J. D. Griffiths, V. Knight, and I. Komenda. Bed management in a Critical Care Unit. *IMA Journal of Management Mathematics*, 24(2):137–153, Jan. 2012. doi: 10.1093/imaman/dpr028.

[34] F. Guerriero and R. Guido. Operational research in the management of the operating theatre: a survey. *Health care management science*, 14(1): 89–114, Mar. 2011. doi: 10.1007/s10729-010-9143-6.

[35] A. Guinet and S. Chaabane. Operating theatre planning. *International Journal of Production Economics*, 85(1):69–81, July 2003. doi: 10.1016/S0925-5273(03)00087-2.

[36] E. Hans, G. Wullink, M. van Houdenhoven, and G. Kazemier. Robust surgery loading. *European Journal of Operational Research*, 185(3):1038–1050, Mar. 2008. doi: 10.1016/j.ejor.2006.08.022.

[37] 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.

[38] W. M. Hirsch and G. B. Dantzig. The fixed charge problem. *Naval Research Logistics Quarterly*, 15(3):413–424, 1968.

[39] J. Hooker and G. Ottoson. Logic-based benders decomposition. *Mathematical Programming*, 96:33–60, 2003.

[40] 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, Dec. 2012. doi: 10.1057/hs.2012.18.

[41] P. J. H. Hulshof, R. J. Boucherie, E. W. Hans, and J. L. Hurink. Tactical resource allocation and elective patient admission planning in care processes. *Health care management science*, 16(2):152–66, June 2013. doi: 10.1007/s10729-012-9219-6.

[42] R. L. Jackson. The Business of Surgery. *Health Management Technology*, 23(7):20–22, 2002.

[43] A. Jebali, A. B. Hadj Alouane, and P. Ladet. Operating rooms scheduling. *International Journal of Production Economics*, 99(1-2):52–62, Jan. 2006. doi: 10.1016/j.ijpe.2004.12.006.

[44] D. G. Kendall. Stochastic Processes Occurring in the Theory of Queues and their Analysis by the Method of the Imbedded Markov Chain. *The Annals of Mathematical Statistics*, 24(3):338–354, Sept. 1953. ISSN 0003-4851. doi: 10.1214/aoms/1177728975.

[45] A. J. Kleywegt, A. Shapiro, and T. Homem-de Mello. The Sample Average Approximation Method for Stochastic Discrete Optimization. *SIAM Journal on Optimization*, 12(2):479–502, Jan. 2002. doi: 10.1137/S1052623499363220.

[46] A. W. J. Kolen, J. K. Lenstra, C. H. Papadimitriou, and F. C. R. Spieksma. Interval scheduling: A survey. *Naval Research Logistics*, 54(5):530–543, 2007. doi: 10.1002/nav.20231.

[47] M. Krishnamoorthy, A. Ernst, and D. Baatar. Algorithms for large scale shift minimisation personnel task scheduling problems. *European Journal of Operational Research*, 219(1):34–48, 2012. doi: 10.1016/j.ejor.2011.11.034.

[48] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics*, 2:83–97, Feb. 1955. doi: 10.1002/nav.20053.

[49] A. Macario, T. S. Vitez, B. Dunn, and T. McDonald. Where Are the Costs in Perioperative Care? *Anesthesiology*, 83:1138–1144, 1995.

[50] J. MacKinnon. An algorithm for the generalized transportation problem. *Regional Science and Urban Economics*, 5(4):445–464, 1975.

[51] J. M. Magerlein and J. B. Martin. Surgical demand scheduling: a review. *Health services research*, 13(4):418–33, Jan. 1978.

[52] A. Marazzi, F. Paccaud, C. Ruffieux, and C. Beguin. Fitting the distributions of length of stay by parametric models. *Medical care*, 36 (6):915–927, 1998.

[53] I. Marques, M. E. Captivo, and M. Vaz Pato. An integer programming approach to elective surgery scheduling. *OR Spectrum*, 34(2):407–427, Dec. 2011. doi: 10.1007/s00291-011-0279-7.

[54] I. Marques, M. E. Captivo, and M. Vaz Pato. Scheduling elective surgeries in a Portuguese hospital using a genetic heuristic. *Operations Research for Health Care*, 3(2):59–72, June 2014. doi: 10.1016/j.orhc.2013.12.001.

[55] A. Mazier, X. Xie, and M. Sarazin. Scheduling Inpatient Admission Under High Demand of Emergency Patients. In *Proceedings of IEEE Conference on Automation Science and Engineering (CASE)*, pages 792–797. IEEE, 2010. ISBN 9781424454495. doi: 10.1109/COASE.2010.5584679.

[56] N. Meskens, D. Duvivier, and A. Hanset. Multi-objective operating room scheduling considering desiderata of the surgical team. *Decision Support Systems*, 55(2):650–659, May 2013. doi: 10.1016/j.dss.2012.10.019.

[57] D. Min and Y. Yih. Scheduling elective surgery under uncertainty and downstream capacity constraints. *European Journal of Operational Research*, 206(3):642–652, Nov. 2010. doi: 10.1016/j.ejor.2010.03.014.

[58] OECD. Health spending (indicator), 2014. URL `http://data.oecd.org/healthres/health-spending.htm`. Accessed on 29 July 2014.

[59] A. Orden. The transhipment problem. *Management Science*, 2(3):276–285, 1956.

[60] I. Ozkaraham. Allocation of Surgeries to Operating Rooms by Goal Programing. *Journal of Medical Systems*, 24(6):339–378, 2000. doi: 10.1023/A:1005548727003.

[61] B. K. Pagnoncelli, S. Ahmed, and A. Shapiro. Sample Average Approximation Method for Chance Constrained Programming: Theory and Applications. *Journal of Optimization Theory and Applications*, 142(2):399–416, Mar. 2009. doi: 10.1007/s10957-009-9523-6.

[62] U. Pferschy and J. Schauer. The maximum flow problem with disjunctive constraints. *Journal of Combinatorial Optimization*, 26:109–119, 2013. doi: 10.1007/s10878-011-9438-7.

[63] 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, Mar. 2008. doi: 10.1016/j.ejor.2006.03.059.

[64] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2012. URL `http://www.R-project.org/`. ISBN 3-900051-07-0.

[65] T. M. Range, R. M. Lusby, and J. Larsen. A column generation approach for solving the patient admission scheduling problem. *European Journal of Operational Research*, 235(1):252–264, May 2014. doi: 10.1016/j.ejor.2013.10.050.

[66] A. Riise and E. K. Burke. Local search for the surgery admission planning problem. *Journal of Heuristics*, 17(4):389–414, Sept. 2011. doi: 10.1007/s10732-010-9139-x.

[67] R. Schmidt, S. Geisler, and C. Spreckelsen. Decision support for hospital bed management using adaptable individual length of stay estimations and

shared resources. *BMC medical informatics and decision making*, 13(1):3, Jan. 2013. doi: 10.1186/1472-6947-13-3.

[68] J. A. M. Schreuder. Combinatorial aspects of construction of competition dutch professional football leagues. *Discrete Applied Mathematics*, 35: 301–312, 1992.

[69] A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming*. Society for Industrial and Applied Mathematics, Jan. 2009. ISBN 978-0-89871-687-0. doi: 10.1137/1.9780898718751.

[70] D. P. Strum, J. H. May, and L. G. Vargas. Modeling the uncertainty of surgical procedure times: comparison of log-normal and normal models. *Anesthesiology*, 92(4):1160–7, Apr. 2000.

[71] M. Sun. The transportation problem with exclusionary side constraints and two branch-and-bound algorithms. *European Journal of Operational Research*, 140:629–647, 2002.

[72] B. G. Thomas, S. Bollapragada, K. Akbay, D. Toledano, P. Katlic, O. Dulgeroglu, and D. Yang. Automated Bed Assignments in a Complex and Dynamic Hospital Environment. *Interfaces*, 43(5):435–448, Oct. 2013. doi: 10.1287/inte.2013.0701.

[73] 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. *OR Spectrum*, 36(3):585–605, Apr. 2013. doi: 10.1007/s00291-013-0323-x.

[74] J. M. van Oostrum, E. Bredenhoff, and E. W. Hans. Suitability and managerial implications of a Master Surgical Scheduling approach. *Annals of Operations Research*, 178(1):91–104, Aug. 2010. doi: 10.1007/s10479-009-0619-z.

[75] 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, Oct. 2010. doi: 10.1057/jors.2010.141.

[76] W. Vancroonenburg. Red-Blue Transportation Problem: problem instances, April 2013. URL `http://allserv.kahosl.be/~wimvc/rbtp/rbtp-instances.zip`. Last accessed on August 4th, 2014.

[77] W. Vancroonenburg and T. Wauters. A late acceptance metaheuristic for the machine reassignment problem. In *25th European Conference on*

Operational Research (EURO2012), page 244, Vilnius, Lithuania, July 2012. Abstract.

[78] W. Vancroonenburg, P. De Causmaecker, and G. Vanden Berghe. A study of decision support models for online patient-to-room assignment planning. *Annals of Operations Research*, 2013. doi: 10.1007/s10479-013-1478-1. Available online.

[79] W. Vancroonenburg, F. Della Croce, D. Goossens, and F. C. R. Spieksma. The Red-Blue transportation problem. *European Journal of Operational Research*, 237(3):814–823, 2014. doi: 10.1016/j.ejor.2014.02.055.

[80] J. Verstichel and G. Vanden Berghe. *Logistic Management*, chapter A late acceptance algorithm for the lock scheduling problem, pages 457–478. Number 5. Springer Berlin / Heidelberg, 2009.

[81] J. M. Vissers, I. J. Adan, and N. P. Dellaert. Developing a platform for comparison of hospital admission systems: An illustration. *European Journal of Operational Research*, 180(3):1290–1301, Aug. 2007. doi: 10.1016/j.ejor.2006.04.034.

[82] Z. Zhao and X. Li. Scheduling elective surgeries with sequence-dependent setup times to multiple operating rooms using constraint programming. *Operations Research for Health Care*, 3(3):160–167, June 2014. doi: 10.1016/j.orhc.2014.05.003.

Curriculum Vitae

Wim Vancroonenburg
Born on December 13, 1987 in Sint-Niklaas, Belgium.

Education & experience

2010 - present	PhD in Engineering Science: Computer Science KU Leuven
2009 - present	Research associate at CODeS research group, Department of Computer Science, Technology Campus Ghent, KU Leuven
2008 - 2009	MSc in Industrial Sciences: Electronics/ICT, option ICT KAHO Sint-Lieven
2005 - 2008	BSc in Industrial Sciences: Electronics/ICT, option ICT KAHO Sint-Lieven

List of publications

Articles in internationally reviewed academic journals

Wauters, T., Kinable, J., Smet, P., **Vancroonenburg, W.**, Vanden Berghe, G., Verstichel, J. (2015). The Multi-Mode Resource-Constrained Multi-Project Scheduling Problem: The MISTA 2013 challenge. Journal of Scheduling. doi: 10.1007/s10951-014-0402-0 (available online)

Vancroonenburg, W., Della Croce, F., Goossens, D., Spieksma, F. (2014). The Red-Blue Transportation Problem. *European Journal of Operational Research*, 237 (3), 814-823.

Garraffa, M., **Vancroonenburg, W.**, Salassa, F., Vanden Berghe, G., Wauters, T. (2014). The one-dimensional cutting stock problem with sequence dependent cut losses. *International Transactions in Operational Research.*

Vancroonenburg, W., Verstichel, J., Tavernier, K., Vanden Berghe, G. (2014). Automatic air cargo selection and weight balancing: A mixed integer programming approach. *Transportation Research E, Logistics and Transportation Review*, 65, 70-83.

Vancroonenburg, W., De Causmaecker, P., Vanden Berghe, G. (2013). A study of decision support models for online patient-to-room assignment planning. *Annals of Operations Research.*

Bilgin, B., Demeester, P., Misir, M., **Vancroonenburg, W.**, Vanden Berghe, G. (2012). One hyperheuristic approach to two timetabling problems in health care. *Journal of Heuristics*, 18 (3), 401-434.

Wauters, T., **Vancroonenburg, W.**, Vanden Berghe, G. (2012). A guide-and-observe hyper-heuristic approach to the Eternity II puzzle. *Journal of Mathematical Modelling and Algorithms*, 11 (3), 217-233.

Papers at international scientific conferences and symposia, published in full in proceedings

Vancroonenburg, W., De Causmaecker, P., Spieksma, F., Vanden Berghe, G. (2013). Scheduling elective patient admissions considering room assignment and operating theatre capacity constraints. In Yeomans, J. (Ed.), Montemanni, R. (Ed.), Nordlander, T. (Ed.), *Lecture Notes in Management Science*: Vol. 5. International Conference on Applied Operational Research (ICAOR'13). Lisbon, Portugal, 29-31 July 2013 (pp. 153-158). Vancouver, Canada: Tadbir Operational Research Group Ltd.

Vancroonenburg, W., De Causmaecker, P., Vanden Berghe, G. (2012). Patient-to-room assignment planning in a dynamic context. *Proceedings of the 9th International Conference on the Practice and Theory of Automated Timetabling.* International Conference on the Practice and Theory of Automated Timetabling (PATAT2012). Son, Norway, 28-31 August 2012 (pp. 193-208).

Misir, M., **Vancroonenburg, W.**, Verbeeck, K., Vanden Berghe, G. (2011). A selection hyper-heuristic for scheduling deliveries of ready-mixed concrete. In Di Gaspero, L. (Ed.), Schaerf, A. (Ed.), Stützle, T. (Ed.), *Proceedings of the 9th Metaheuristics International Conference.* Metaheuristics International Conference 2011 (MIC2011). Udine, Italy, 25-28 July 2011 (pp. 289-298).

Verstichel, J., **Vancroonenburg, W.**, Souffriau, W., Vanden Berghe, G. (2011). A mixed integer programming approach to the aircraft weight and balance problem. In Żak, J. (Ed.), *Procedia - Social and Behavioral Sciences*: Vol. 20. The State of the Art in the European Quantitative Oriented Transportation and Logistics Research – 14th Euro Working Group on Transportation & 26th Mini Euro Conference & 1st European Scientific Conference on Air Transport. Poznań, Poland, 6-9 September 2011 (pp. 1051-1059) ELSEVIER.

Wauters, T., **Vancroonenburg, W.**, Vanden Berghe, G. (2010). A two phase hyper-heuristic approach for solving the Eternity II puzzle. *Proceedings of the International Conference on Metaheuristics and Nature Inspired Computing.* International Conference on Metaheuristics and Nature Inspired Computing. Djerba Island, Tunisia, 27-31 October 2010 (art.nr. 164) (pp. 1-3).

Meeting abstracts, presented at international scientific conferences and symposia, published or not published in proceedings or journals

Vancroonenburg, W., Smet, P., Vanden Berghe, G. (2014). A two phase heuristic approach to muti-day surgical case scheduling considering generalized resource constraints and desiderata. *Operational Research Applied to Health Services (ORAHS2014).* Lisbon, Portugal, 20-25 July 2014.

Salassa, F., Grosso, A., **Vancroonenburg, W.** (2013). Searching for a maximal dominating cycle in undirected graphs. *European Conference on Operational Research (EURO/INFORMS).* Rome, Italy, 1-4 July 2013.

Vancroonenburg, W., Wauters, T. (2013). Extending the late acceptance metaheuristic for multi-objective optimization. *Multidisciplinary International Scheduling conference: Theory & Applications (MISTA2013).* Ghent, 27-30 August 2013.

Garraffa, M., Salassa, F., **Vancroonenburg, W.**, Vanden Berghe, G., Wauters, T. (2013). A computational study of a cutting stock problem with sequence dependent cut losses. *Multidisciplinary International Scheduling conference: Theory & Applications (MISTA2013).* Ghent, 27-30 August 2013.

Salassa, F., Wauters, T., **Vancroonenburg, W.**, Della Croce, F., Vanden Berghe, G. (2012). The Eternity II puzzle: a matheuristic approach. *International Symposium on Combinatorial Optimization.* Athens, Greece, 17-21 April 2012.

Vancroonenburg, W., De Causmaecker, P., Spieksma, F., Vanden Berghe, G. (2012). Patient-to-room assignment planning in a dynamic context. *Operational Research Applied to Health Services (ORAHS2012).* Twente, The Netherlands, 15-20 July 2012.

Vancroonenburg, W., Wauters, T. (2012). A late acceptance metaheuristic for the machine reassignment problem. *European Conference on Operational Research (EURO2012).* Vilnius, Lithuania, 8-11 July 2012.

Smet, P., **Vancroonenburg, W.**, Vanden Berghe, G. (2011). Automated scheduling at a joinery site. *Proceedings of the 5th Multidisciplinary International Conference on Scheduling: Theory and Applications.* Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA2011). Phoenix, Arizona, USA, 9-11 August 2011, 549-552.

Vancroonenburg, W., Spieksma, F., Vanden Berghe, G., De Causmaecker,

P. (2011). A study of the patient assignment problem with gender constraints. *Operational Research Applied to Health Services (ORAHS2011).* Cardiff, United Kingdom, 24-29 July 2011, Abstract No. 79.

Vancroonenburg, W., Misir, M., Bilgin, B., Demeester, P., Vanden Berghe, G. (2010). A hyper-heuristic approach for assigning patients to hospital rooms. *Proceedings of the 8th International Conference on the Practice and theory of automated timetabling.* Practice and Theory of Automated Timetabling. Belfast, 10-13 August 2010, 553-555.

Meeting abstracts, presented at international professionally oriented conferences and symposia, published or not published in proceedings or journals

Vancroonenburg, W. (2013). Automated decision-support for hospital admission planning. *iMinds The conference.* Brussels, Belgium, 5 December 2013.

Meeting abstracts, presented at other scientific conferences and symposia, published or not published in proceedings or journals

Vancroonenburg, W., De Causmaecker, P., Spieksma, F., Vanden Berghe, G. (2014). Integrating patient-to-room assignment planning with elective admission scheduling. *Proceedings of the 28th Annual Conference of the Belgian Operations Research Society.* ORBEL. Mons, Belgium, 30-31 January 2014.

Vancroonenburg, W., Della Croce, F., Goossens, D., Spieksma, F. (2013). The Red-Blue Transportation Problem. *Proceedings of the 27th Annual Conference of the Belgian Operations Research Society.* ORBEL. Kortrijk, Belgium, 7-8 February 2013.

Vancroonenburg, W., De Causmaecker, P., Spieksma, F., Vanden Berghe, G. (2012). The patient assignment problem: complexity and planning in a dynamic context. *Proceedings of the 26th Annual Conference of the Belgian Operations Research Society.* ORBEL. Brussels, Belgium, 2-3 February 2012, 102-103.

Wauters, T., **Vancroonenburg, W.**, Salassa, F., Della Croce, F., Vanden Berghe, G. (2012). The Eternity II puzzle: a matheuristic approach. *Proceedings* *of the 26th Annual Conference of the Belgian Operations Research Society.* ORBEL. Bruxelles, 2-3 February 2012.

Vancroonenburg, W., Misir, M., Vanden Berghe, G. (2011). A hyper-heuristic approach for the ready-mixed concrete delivery problem. *Proceedings of the 25th Annual Conference of the Belgian Operations Research Society.* ORBEL. Gent, 10-11 February 2011, 73-74.

Vancroonenburg, W., Wauters, T., Vanden Berghe, G. (2010). A real world 1D stock cutting problem: Exact and heuristic algorithms. *Proceedings of the 24th Annual Conference of the Belgian Operations Research Society.* ORBEL. Liege, 28-29 January 2010.

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
COMBINATORIAL OPTIMISATION AND DECISION SUPPORT
Celestijnenlaan 200A box 2402
B-3001 Leuven