NTNU – Trondheim
Norwegian University of
Science and Technology

Operating Room Scheduling Problem

Considering the uncertain arrivals of the emergency patients and the capacity limits of the pre-op and post-op facilities

Sofie Døving Agdestein

Industrial Economics and Technology Management
Submission date: June 2012
Supervisor: Lars Magnus Hvattum, IØT

Norwegian University of Science and Technology
Department of Industrial Economics and Technology Management

NTNU
Det skapende universitet

MASTERKONTRAKT

- uttak av masteroppgave

1. Studentens personalia

Etternavn, fornavn **Agdestein, Sofie Døving**	Fødselsdato **03. des 1987**
E-post **sofie@agdestein.no**	Telefon **93200400**

2. Studieopplysninger

Fakultet **Fakultet for Samfunnsvitenskap og teknologiledelse**	
Institutt **Institutt for industriell økonomi og teknologiledelse**	
Studieprogram **Industriell økonomi og teknologiledelse**	Hovedprofil **Anvendt økonomi og optimering**

3. Masteroppgave

Oppstartsdato **16. jan 2012**	Innleveringsfrist **11. jun 2012**
Oppgavens (foreløpige) tittel **Operating Room Scheduling Problem** **Including the uncertainty of non-elective patient arrivals**	
Oppgavetekst/Problembeskrivelse The purpose of this thesis is to develop an optimization method for solving the operating room scheduling problem (ORSP), including the uncertainty of non-elective patient arrivals. The ORSP consists of scheduling elective surgeries to specific time periods and reserving the required resources to each patient (such as rooms, human resources and equipment), while minimizing costs. The problem also considers the capacity constraints of the the pre-op and post-op facilities. The ORSP is to be solved by the use of a stochastic model, looking at the possible different scenarios of non-elective patient arrivals. The model will be implemented in XpressMP.	
Hovedveileder ved institutt **Førsteamanuensis Lars Magnus Hvattum**	Medveileder(e) ved institutt
Merknader **1 uke ekstra p.g.a påske.**	

4. Underskrift

Student: Jeg erklærer herved at jeg har satt meg inn i gjeldende bestemmelser for mastergradsstudiet og at jeg oppfyller kravene for adgang til å begynne oppgaven, herunder eventuelle praksiskrav.

Partene er gjort kjent med avtalens vilkår, samt kapitlene i studiehåndboken om generelle regler og aktuell studieplan for masterstudiet.

Mysore, India, 13.01.12
Sted og dato

Sofie D. Agdestein
Student

[illegible]
Hovedveileder

Originalen lagres i NTNUs elektroniske arkiv. Kopi av avtalen sendes til instituttet og studenten.

Sammendrag

Denne masteroppgaven legger fram en løsningsmetode for problemet *tidsplanlegging for operasjonsrom* (ORSP) med to typer etterspørsel: operasjoner av elektive pasienter og operasjoner av et ukjent antall akuttpasienter. Problemet går ut på å lage en plan over når, og i hvilket rom, de elektive pasientene skal opereres, samtidig som de totale kostnader minimeres. Det tas hensyn til usikkerheten i forhold til ankomstene av akuttpasienter og kapasitetsbegrensningene i de pre- og postoperative fasiliteter. Problemet er modellert med bruk av flerstegs stokastisk programmering, og de mulige hendelsesforløpene av akuttpasientankomster vises i scenariotrær. To typer recourse beslutninger blir tatt i hvert steg. En elektiv pasient som er planlagt operert i tidsperioden til steget kan enten bli utsatt én tidsperiode (type 1 beslutning) eller få forandret operasjonsrom innen samme tidsperiode (type 2 beslutning). I tillegg må alle akuttpasientene som ankommer fordeles på operasjonsrom i hvert steg. Modellen er implementert i Xpress$^{\mathrm{MP}}$. To heuristikker er anvendt på problemet: heuristikken "fikser og relakser" ("fix and relax") og en forbedringsalgoritme. Studien av det implementerte problemet viser at det å inkludere usikkerhet ved hjelp av den presenterte flerstegsmodellen er nyttig for problemer som representeres av opp til 8 scenarier. Modellen yter best når kun type 1 beslutninger tillates i hvert steg. For testinstansene som er benyttet er det en liten verdi av å inkludere kapasitetsbegrensninger i pre-op og post-op.

Abstract

This thesis proposes a solution approach to the operating room scheduling problem (ORSP) with two types of demand for surgery: known elective demand and uncertain emergency demand. The ORSP consists of scheduling elective surgeries to an operating room and a time period, while minimizing costs. The uncertainty regarding emergency patient arrivals and the capacity constraints of the pre-op and post-op facilities are taken into account. The problem is modeled using multi-stage stochastic programming, and the dynamics of the emergency patient arrival process are shown using a scenario tree structure. Two types of recourse decisions are allowed in each stage; a scheduled elective patient may be postponed one time period (type 1), or the operating room can be changed for the elective patient within the same time period (type 2). In addition, the emergency patients arriving must be allocated rooms in each stage. The model is implemented in Xpress$^{\text{MP}}$. Two heuristics are applied to the model: fix and relax and an improvement algorithm. The computational study shows that including the uncertainty by using the multi-stage model presented is beneficial for problems represented by up to 8 scenarios. The model performs the best when only allowing recourse decisions of type 1. For the test instances used, including the pre-op and post-op capacity constraints seem to be of a small value.

Preface

This master's thesis is the final work accomplished to achieve a Master of Science degree with specialization in Applied Economics and Optimization at the Department of Industrial Economics and Technology Management at the Norwegian University of Science and Technology (NTNU) during the spring of 2012.

I would like to thank my supervisor Lars Magnus Hvattum at NTNU for good guidance and constructive discussions throughout the semester.

Trondheim, June 5, 2012

Sofie D. Agdestein

Sofie Døving Agdestein

Table of contents

List of tables

List of figures

1. Introduction

Good management of the health services in Norway is becoming increasingly important. Calculated per capita, Norway has the second highest health expenditures in the world, only after the USA [23]. The expenditures have increased with 84% during the last decade, and as a share of the Norwegian GDP, they have been varying around 9% in the same period [24]. The corresponding mean of the OECD countries was 8.6% [23]. Two reasons for the large growth are that both the demand and the costs of health care increase [18]. The growing demand could partly be due to a growing aging population [23], partly be due to technological developments that have broadened the scope of surgical interventions [11], and partly be due to that the population in general has more money to spend [18].

About half of the Norwegian health expenditures are used on medical treatment [24]. Within the hospital, the operating theatre is one of the most costly resources. The management of this section will make a major impact on the performance of the hospital as a whole [6]. Good planning procedures can reduce the costs significantly, as the wasted time of scarce and expensive resources is minimized, and the productivity and effectivity are increased. In addition to the economical aspect, the satisfaction level of the patients and employees can increase. Good planning enforces shorter waiting times for the patients, a higher level of efficiency resulting in more patients treated, and increased predictability of the surgery waiting time. The amount of overtime working hours for the staff can be minimized, and the level of skill of the personnel can be better made use of.

To enforce good management of the operating theatre, the operating room planning and scheduling problem is of great importance. This problem can be split into two parts: a long term planning problem and a more short term scheduling problem. The operating room *planning* problem is about making supply meet demand. The hospital needs to make sure that they hire the right amount of surgeons and nurses, and that they have the right equipment and enough operating rooms [6].

The operating room *scheduling* problem is about how the available resources are allocated. The problem can be split into *advance* scheduling and *allocation* scheduling: Advance scheduling is the process of fixing a surgery date for a patient, allocation scheduling determines the operating room, the resource assignment, and the starting time of the procedure on the specific day of surgery (sequencing) [6]. The problem is subject to a large amount of uncertainty. Important uncertainties are related to the demand (emergency patients arrive unexpectedly and the treatment requirements of the individual patients can vary a lot [5]), the resource availability (machines fail and surgeons may be away), and the duration of the surgeries.

This thesis addresses the problem of scheduling dates and operating rooms for elective surgeries, referred to as the operating room scheduling problem, ORSP. The problem defined is positioned in between the advance scheduling and the allocation scheduling.

Most of the literature on operating room planning and scheduling focus on either elective or non-elective patients only. General optimization models on operating room planning and scheduling exist, but few include the uncertainty of non-elective patient arrivals. In the ORSP defined in this thesis, the issue is addressed using stochastic programming. The stochastic programming solution provides a suggested plan of when and where to schedule elective patients surgeries. This is done under capacity constraints of the pre- and post facilities of the operating room, and the objective of the problem is to minimize costs.

Other relevant objectives for the ORSP could be balancing and maximizing the utilization level of the resources (avoiding expensive under-utilization, but still keeping buffers to avoid extra costs due to uncertainty), and minimizing the patients waiting time for surgeries. The reason for choosing a financial objective is that financial numbers are easily quantifiable and often more available and intuitive to operate with. Also, any cost savings can be invested in improving the other objectives.

In real life, the hospitals tend to overbook the capacity of the ORs, as it is a common understanding that it is better to cancel a few surgeries than risking unused resources [22]. Assigning a particular OR and the corresponding set of resources to each surgery in a relatively early scheduling phase, before the final scheduling into time slots per day is done, makes it more likely that the required resources are available for the elective patients, and that the surgeries are performed on time. The solution of the ORSP is intended to be a decision support tool, giving a guidance on when to schedule the patients and on how much capacity should be reserved for the non-elective patients.

This thesis is divided into 11 chapters. The description of the ORSP is given in Chapter 2. Then, in Chapter 3 some existing literature on the subject is presented. Theory relevant for the problem is given in Chapter 4. Then, in Chapters 5 and 6, the deterministic and stochastic formulations of the problem are presented. Chapter 7 provides additional information about the heuristics applied to the problem. Chapter 8 presents the test instances used in the computational study, and Chapter 9 describes the implementation of the model and the computational study done. The conclusions are summed up in Chapter 10. Possible further work is discussed in Chapter 11.

2. Description of the ORSP

In this chapter, the fundamentals of the problem addressed in this thesis will be presented.

The aim of the problem presented is to provide hospitals a tool that can aid them in making a surgery schedule for the elective patients that will reduce the total expected costs to a minimum. The problem consists of scheduling the elective patients to time periods and rooms. The decisions must take into consideration the uncertainty regarding the emergency patient arrivals and the constraints regarding available resources. These constraints are time limits of the operating rooms and the capacity limits of the pre-op and post-op facilities.

The scheduling of elective surgeries in a hospital is done for a definite planning horizon, split into time periods. In a medium term perspective, one planning period could for example be a day, and one planning horizon could be five days, Monday to Friday. Each of the operating rooms in the hospital is listed with a total available regular capacity per time period. If the regular capacity is exceeded, overtime will occur. The overtime is split into time intervals of different price ranges, and the total overtime can not exceed the total available capacity of the price ranges.

The capacity is shared between the two patient groups: elective and emergency patients. *Elective* patients are already known patients, for which the surgery can be planned in advance. *Non-elective* patients are unexpected patients, with stochastic arrivals. All non-elective patients are treated as emergencies, and they have to be taken care of on the day of arrival. The expressions "emergency patients" and "non-elective patients" are used interchangeably in this thesis.

The number of elective cases to be scheduled is known and not a subject to uncertainty. The surgeries need to be performed on one of the days that is possible for the particular patient. The first day possible is the *release period*, when everything is ready for the surgery. This can depend on for example hospitalization date or date of medical test delivery. An elective patient can also be postponed until the next planning period.

Each surgery requires a certain amount of time to be done. This includes all the time the OR is occupied with activities related to the respective surgery; such as the time needed for preparing the machines and tools, the time needed to perform the surgery and the time needed to clean the OR. Every surgery must be allocated to one of the ORs defined as suitable for the patient. The ORs are connected to specific sets of resources, such as personnel and machines. A set of costs is associated with each elective case, time period, and OR. These costs represent hospitalization costs, penalties for waiting time, preferences, or other aspects beneficial to include.

The patients move through three stages: the preoperative, perioperative, and postoperative stage. Each patient occupies a spot in the pre-op facility, with limited capacity, on the day of surgery. In the *perioperative* stage the patients are operated on. After the surgery, in the *postoperative* stage, most of the patients are moved to the postanaesthesia care unit (PACU). The patients requiring intensive care are moved to the intensive care unit (ICU). Each patient spends a given number of days at the post-op, either in the ICU or the PACU. Both units have limited capacities. Costs are associated with the violation of the capacity limits, as it is undesirable, or sometimes impossible, to exceed these capacities.

In each time period, three types of decisions must be made. First, the emergency patients that arrive need to be allocated to ORs. This can lead to change of plans for the originally scheduled elective patients. The elective patients may be postponed one time period, resulting in a day extra at pre-op, or they may be changed OR for within the same time period. These decisions can lead to extra costs.

The objective of the problem is to minimize costs. The total costs consists of the costs of performing the elective cases, any extra costs due to change of plans for the originally scheduled patients, overtime costs, and capacity violation costs for the pre-op and post-op facilities.

3. Literature review

In this chapter, an overview of relevant literature for the work is presented. Section 3.1 provides a literature review of the general OR scheduling problem. Next, in Section 3.2, literature on OR scheduling problems including non-elective patients is presented. Last, in Section 3.3, some work done on the OR scheduling problems including the capacity constraints of the pre-op and post-op facilities is presented.

3.1 OR scheduling

The operating theater generates very high costs for the hospital. Together with the fact that the administration of this unit has a large influence on the patients health and satisfaction level, it is of great importance to focus on good management and making a good OR scheduling plan. However, it can be hard due to conflicting priorities and preferences, and due to the scarcity of costly resources. Also, the demand is largely uncertain, due to the arrivals of emergency patients. The planning of surgical operations is therefore highly complex, and it has been written a lot of literature on the subject in the last decades. Cardoen, Demeulemeester and Beliën [6] provides a review on recent research within the field.

The review is from 2009 and includes 124 references to relevant literature published in or after 2000. These are structured into 6 descriptive fields, depending on what they have their main focus on: *Patient characteristics* (elective or non-elective, inpatient or outpatient), *performance measures* (waiting time, patient deferral, utilization, make-span, financial value, preferences or throughput), *decision delineation* (what type of decision has to be made and whether this decision applies to a medical discipline, a surgeon or a patient (type)), *research methodology* (information on the type of analysis that is performed and the solution evaluation techniques and applications), *uncertainty* (stochastic versus deterministic approaches to how arrival or surgery duration uncertainty is handled), and *applicability of research* (information on the testing of research and its implementations in practice).

3.2 OR scheduling including non-elective patients

Most of the literature written on patient planning and scheduling regards the elective patients only. Cardoen et al. [6] points out that the large degree of uncertainty regarding the non-elective patients is the main reason why operating room scheduling urges other scheduling methodologies than the machine scheduling procedures developed for industrial systems, and that it could be beneficial to do more studies regarding both elective and non-elective patients.

Wullink et al. [21] look at how to best reserve operating room capacity for emergency surgery. They present and compare two basic approaches: dedicating specific operating room to non-elective patients, and evenly reserving capacity in all elective ORs. The real situation was modeled using discrete-event simulation, and they found that the best way of reserving capacity for emergency surgery, regarding responsiveness, amount of overtime and the overall utilization of the ORs, was to spread the capacity over multiple rooms.

Bhattacharyya et al.[3] also discuss the effects of reserving specific ORs for emergency patients only. Their study focuses on the orthopaedic unit of a hospital, and the concluding recommendation was that hospitals and orthopaedic trauma services should keep an open OR reserved for orthopaedic trauma.

Bowers and Mould [5] look at how the large uncertainty in surgery demand, due to emergency patients, influences the utilization of the orthopaedic trauma theatre. Simulations and approximations, examined as an alternative to the full simulation, were used to study the balance between maximizing the utilization of the theatre and avoiding lack of quality or too many overruns. They found that it appears that if elective patients are willing to accept a possibility of their treatment being cancelled, substantially greater throughputs could be achieved, and that the approximated simpler model offers reasonable accuracy.

Krempels and Panchenko [14] look at how the high complexity of the planning of surgical operations can be dealt with. To handle the new information appearing as the uncertain data is revealed (emergency cases occur), they suggest a semi-automated dialog-based system, involving a human planner in the scheduling activity. The planner is meant to act as a "sensor" to identify changes as they occur and integrate his knowledge and decision-making competence into the planning process. They also discuss heuristics suited to create proposals for surgery schedule.

Lamiri et al. [15] describe a stochastic model for OR planning with both elective and emergency surgery demand. A stochastic mathematical programming model is suggested, and a Monte Carlo optimization method combining Monte Carlo simulation and Mixed Integer Programming is proposed to model the uncertainties. The problem consists of determining a plan that specifies the set of elective cases that would be performed in each period over a planning horizon of one or two weeks, and the objective is to minimize costs. The suggested model served as a starting point for the model presented in this thesis.

3.3 OR scheduling with capacity constraints for the post-operative instances

There has been written little literature focusing on OR and post-op together, and the interaction with pre-op does not seem to have been addressed systematically up to now [16]. Gupta [12] describes some commonly occurring operations management problems faced by the managers of surgical suites, one of which related to the capacities of the pre-op and post-op facilities. The goal of the article is to identify open challenges to motivate further research. A model is presented for surgery booking control, which takes into account limited capacity of a critical downstream resource; the post-op facilities.

Pham and Klinkert [16] present a new surgical scheduling approach, using an extension of the Job Shop scheduling problem called multi-mode blocking job shop (MMBJS). They formulate the MMBJS as a mixed integer linear programming problem and discuss the use of the model for scheduling elective and emergency surgeries. They also point out the importance of connecting the different surgical stages (pre-op and post-op) when scheduling any surgical case.

4. Theory

This chapter describes relevant theory used in the work of this thesis. Section 4.1 gives an introduction to stochastic programming, Section 4.2 discusses how stochastic models can be evaluated, and Section 4.3 presents the two heuristics used in the solution process of the ORSP.

4.1 Stochastic programming

Stochastic programming is a framework for modeling optimization problems that involve uncertainty. First in this section, uncertainty within the optimization models will be commented on, then a stochastic programming approach to handle the uncertainty is discussed; two-stage and multi-stage recourse models.

4.1.1 Uncertainty in optimization models

When using *deterministic programming*, all the parameters are assumed to be known and fully predictable. In practical situations, however, many of the parameters are often uncertain. The uncertainty can for example lie in in prices, demand, costs, weather, and technology. The deterministic optimization methods fail to take the impact of uncertainty in the problem into account, and the solutions obtained can therefore be of limited value.

Stochastic optimization models treat the uncertain parameters as random variables, with distribution functions representing possible outcomes and their respective probabilities. The distributions may be continuous or discrete, resulting in problems of different complexity. *Recourse models* are examples of stochastic optimization models. In recourse models, stages are introduced. Stages in time, where new information becomes available and decisions can be made, allow for a more realistic representation of real problems. In this way, corrective actions can be done as a response to the new information. This method will be described more thoroughly in the next sections.

4.1.2 Two-stage stochastic programming models

Basic properties In a two-stage stochastic programming problem, the decisions are made in two stages. The (vector) value of a random variable is revealed in between the two stages. The second stage decision is called the recourse decision and is used to compensate for any negative effects or exploiting positive effects that resulted from the first stage decision. The goal of a twostage model is to identify a first stage solution that is well positioned against all possible observations of the random variable. The solution scheme is as follows:

1. The first stage problem is solved; the first stage decisions, x, are determined
2. The random variable, $\widetilde{\omega}$, is observed
3. The recourse problem is solved; the second stage decisions, y, are made

Recourse formulation The general form of the two-stage stochastic programming problem is the recourse formulation, shown in (4.1) and (4.2).

$$\begin{aligned} \min \quad & cx + E[h(x, \widetilde{\omega})] \\ \text{s.t.} \quad & Ax \geq b \\ & x \geq 0 \end{aligned} \tag{4.1}$$

where

$$\begin{aligned} h(x, \omega) = \min \quad & g_\omega y \\ \text{s.t.} \quad & W_\omega y \geq r_\omega - T_\omega x \\ & y \geq 0 \end{aligned} \tag{4.2}$$

A part of the objective function in (4.1) is dependent on an unknown variable, $\widetilde{\omega}$. The set of possible scenarios is represented by Ω, and the different scenarios are represented by ω, $\omega \in \Omega$. The problem in (4.2) can be called the second stage problem, subproblem, or recourse subproblem [13]. After the unknown variable is known, the subproblem for the relevant scenario is solved to optimality.

Scenario formulation In the scenario formulation, each possible scenario is formulated as a deterministic problem, and non-anticipativity constraints are added to ensure that the information structure associated with the decision process is honored. For a single scenario, the deterministic problem is as given in (4.3).

$$\begin{aligned} \min \quad & cx_\omega + g_\omega y_\omega \\ \text{s.t.} \quad & T_\omega x_\omega + W_\omega y_\omega \geq r_\omega \\ & x_\omega, y_\omega \geq 0. \end{aligned} \tag{4.3}$$

When each scenario is weighted in the objective function with a probability factor p_ω, and extra constraints are added to ensure that the x_ω equals the original decision of x, we get the scenario formulation, shown in (4.4).

$$\begin{aligned} \min \ & \sum_{\omega\in\Omega} (cx_\omega + g_\omega y_\omega) p_\omega \\ \text{s.t.} \ & T_\omega x_\omega + W_\omega y_\omega \geq r_\omega \\ & x_\omega - x = 0 \ \ \forall \omega \in \Omega \\ & x_\omega, y_\omega \geq 0. \end{aligned} \tag{4.4}$$

Scenario trees Scenario trees are often used to characterize the uncertainty in a stochastic optimization problem. A scenario tree depicts how the possible realizations of an uncertain parameter lead to different scenarios, and how the scenarios are related to each other. An example of a three stage scenario tree is shown in Figure 4.1.

To be able to create a scenario tree for a problem, one needs to assume that the probability distribution of the random quantities affecting the problem solution can be described as a discrete distribution with a finite number of possible outcomes. This may be an approximation of the problem, as the decisions often could be taken at any point in time, and the random variables often are better represented by a continuous distribution.

The discrete distribution can be represented through an event tree with nodes associated with the realizations of the stochastic quantities. The nodes represent information states, specific points in time when a realization of the random process becomes known and a subsequent decision is taken. The structure of the event tree gives information about the information arrival process. The root node represents the initial decision stage. In this stage, no observations of the stochastic parameters have been made yet. A path from the root to a leaf node of the event tree represents a scenario. The probability of each scenario is the product of the conditional probabilities of visiting each of the nodes on the path. The end leaves represent the possible futures of the decision problem.

To each node of the event tree one associates a set of constraints, an objective function, and the conditional probability of visiting the node from its parent node in the previous stage.

Using a scenario tree to describe the uncertainty gives the opportunity to formulate the deterministic equivalent problem with implicit or explicit non-anticipativity constraints, in a node formulation or scenario formulation, respectively. These constraints will be explained further in the next paragraphs.

Non-anticipativity In a multi-stage framework information is revealed through time, and the decisions made at a certain stage may only depend on the realizations of the stochastic quantities up to that point and on the decisions previously made. To make sure that the decision sequence honors the information structure, non-anticipativity constraints are added. They make sure that scenarios that share the same history of information until a particular decision stage also have made the same decisions [13]. In other words, they make sure that the actions that must be taken at a specific point in time depend only on the information that is available at that time.

The non-anticipativity constraints may be formulated in explicit or implicit form [2]. In the case of explicit constraints, the event tree is split path-wise. The decision process follows the scenario evolution, and the decision problems are solved locally in every node. The procedure leads to as many dynamic problems as the amount of scenarios. The method is called *scenario formulation*, or "split-variable" form, since the variables given in the implicit form are split into several variables, according to scenarios.

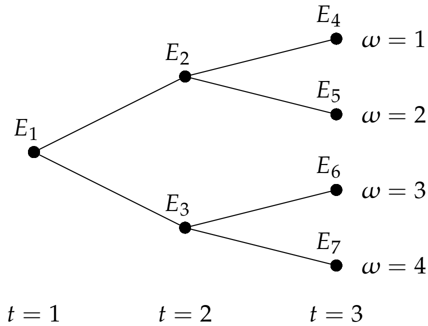
In the case of implicit constraints, a unique vector of decision variables for each node of the tree is introduced. This makes sure that the random coefficients of the problem are properly associated. This is called a *node formulation*.

4.1.3 Multi-stage stochastic programming models

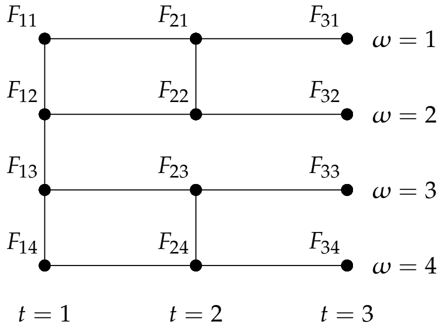
Basic properties In a multi-stage stochastic programming problem, the decisions are made in multiple stages. The "decide - observe - decide" pattern is repeated several times. This may lead to large and complex problems. The uncertain data $\xi_1, \ldots, \xi_T$ is revealed gradually over time, in T periods, and the decisions taken are adapted to this process. As the decision vector x_t may depend on the uncertain data ξ_t, the sequence of decisions is also a stochastic process. The values of the decision vector x_t can only depend on the available information in stage t, and not on the results of future observations. This is a basic requirement of non-anticipativity.

As in the two-stage recourse problems, the multi-stage recourse problem can be written in either a scenario formulation or a node formulation. The scenario formulation is often more intuitive to read and write, but the node formulation can ease the solving process, requiring fewer variables and constraints.

Node formulation Figure 4.1(a) shows an example of a three stage scenario tree with four possible scenarios, ω. Each node is labeled E_n, where n represents the node number. $t(n)$ represents the stage number of the node. Every outcome of the random vector ξ_1, ξ_2 corresponds to a unique path from $E_n, t(n) = 1$ to $E_n, t(n) = 3$ on the scenario tree. The non-anticipativity constraints are formulated in an implicit form. Each node in the scenario tree corresponds to a collection of scenarios at a specific stage [13]. When in node 3, it is not possible to recognize which of the scenarios $\omega = 3$ or $\omega = 4$ will ultimately result.



(a) The structure of the node formulation



(b) The structure of the scenario formulation

Figure 4.1: A three-stage problem

Scenario formulation The structure of the scenario formulation for a three-stage model is illustrated in Figure 4.1(b). Each scenario is represented by a horizontal line. All variables are defined along each scenario, also the ones that are not scenario dependent or the ones that are common for some of the scenarios. The variables are labeled with both a stage index t, and a scenario index ω, then linked according to the illustration with explicit non-anticipativity constraints [19].

4.2 Evaluation of stochastic models

A stochastic approach can result in non-linear problems and large-scale models. Multi-stage programs, in particular, have the reputation of being computationally difficult to solve [9]. Specific solution methods are often required, and a lot of effort and care is needed for estimating acceptable probability distributions of the uncertain parameters. Deterministic models are usually a lot less demanding to handle. The main effort will be put into determining the uncertain parameters. However, the expected value solution, obtained from solving the deterministic problem, can often be insufficient in an uncertain environment.

To decide whether or not it is necessary to use a stochastic model, different evaluation tools have been developed. Two methods will be presented in the following section, Section 4.2.1: *the value of stochastic solution (VSS)* and the *expected value of perfect information (EVPI)*. These concepts were developed for two-stage problems, and are described further in Birge and Louveaux [4]. The notation used is in regards to a minimization problem, since the goal of the problem described in this thesis is to minimize costs. In Section 4.2.2, the corresponding tools to evaluate multi-stage models are discussed.

4.2.1 The value of stochastic solution and the expected value of perfect information

A common and simple method for including uncertainty in the problem solving is solving the *expected value problem (EV)*. In the EV, all the uncertain parameters are replaced by their expected values, and the problem is solved as a deterministic problem.

Using the solution to the EV, the *expected value of the expected value solution (EEV)* can be calculated. The EV solution, obtained from the deterministic problem, is included in the recourse model as the first stage decision. Then, the problem is solved with these variable values fixed for all the different scenarios, calculating associated recourse variables and costs. The EEV is the expected average performance of the deterministic solution under uncertain conditions.

Further, the solution to the EEV can be used to calculate the *value of stochastic solution (VSS)*, as shown in equation (4.5). The VSS evaluates what it is worth to include uncertainty in the model, solving the stochastic *recourse problem (RP)*, rather than solving the corresponding deterministic problem.

$$VSS = EEV - RP \tag{4.5}$$

For a minimization problem, $VSS \geq 0$. This is because the solution of the RP must be equal to or better than the EEV, when solved to optimality. A small VSS indicates that replacing the random variables by their expected values is a good approximation [9].

The *wait-and-see solutions (WS)* can be found by solving each scenario problem in isolation, for each possible outcome of the uncertain parameters. These solutions are the expected solutions if all uncertainty is removed.

The *expected value of perfect information (EVPI)* measures the possible gains from eliminating all the uncertainties from the stochastic model. First, to calculate the $EVPI$, the average of the individual optimal solutions (WS) is calculated based on the probability of each outcome, and represents the average performance in the case of perfect information. Then, the $EVPI$ is computed as the difference between the optimal stochastic solution and the average performance with perfect information, as shown in equation (4.6).

$$EVPI = RP - WS \tag{4.6}$$

The $EVPI$ can be an indication on the maximum amount a decision maker should be willing to pay to receive complete and accurate information about the future.

For a two-stage minimization recourse model, the relations given in equation (4.7) are satisfied.

$$WS \leq RP \leq EEV \tag{4.7}$$

These bounds make it possible to evaluate the potential gains from applying a recourse model.

4.2.2 Evaluation tools for multi-stage models

The VSS and $EVPI$, presented above, were developed for two-stage problems. Escudero et al. [9] have generalized those parameters to the multi-stage case. They discuss which variables that should be fixed in the WS models when calculating the EEV.

For the two-stage case, the EEV was calculated as the WS models with first stage solutions fixed from the deterministic solution. If the same method is followed in the multi-stage case; fixing the first stage variables and letting the variables of the following stages be free to adapt to the performance of the different scenarios, it may happen that the first stage solution in the EV problem performs better than the solution of the RP [9]. The reason is that when the WS models are solved, every scenario is solved as if it was independent from the others. The non-anticipativity constraints are ignored. Therefore, the EEV should be redefined for the multi-stage models, taking into consideration the non-anticipativity constraints. One approach of doing this, described in in Escudero et al. [9], will now be presented.

To adapt the EEV to the multi-stage setting, the *expected result in stage t of using the expected value solution (EEV_t)* is introduced. This value represents the optimal value of the RP model, where the decision variables until stage $t-1$ are fixed at the optimal values obtained in the solution of the EV model. For $t = 1$, we define $EEV_1 = RP$. The relation given in equation (4.8) holds for any multi-stage stochastic minimization program.

$$EEV_{t+1} \geq EEV_t, \qquad t = 1, ..., T-1 \tag{4.8}$$

The corresponding *value of stochastic solution in stage t (VSS_t)* is defined in equation (4.9).

$$VSS_t = EEV_t - RP, \qquad t = 1, ..., T \tag{4.9}$$

The relation given in equation (4.10) holds for any multi-stage stochastic program.

$$0 \leq VSS_t \leq VSS_{t+1}, \qquad t = 1, ..., T-1 \tag{4.10}$$

The sequence of VSS_t represents the cost of ignoring uncertainty until stage t in the decision making in multi-stage models, or the performance of the approximation of the random variables by their expected values up to stage t.

4.3 Heuristics for MIP

Real world MIP problems are often very complex and hard to solve to optimality. To ease the solving process of the problems, different heuristics can be applied. The heuristics can be classified into two types: constructive heuristics and improvement heuristics [1]. The *constructive heuristics* are methods that try to build a feasible solution from scratch. The *improvement heuristics* take an already existing solution (for example found by constructive heuristics) and successively try to improve it.

Heuristics used to construct solutions for a general MIP are among others *LP-and-Fix*, where the LP problem is solved, all the integrals are fixed, and the MIP is solved again, and *Fix and relax*, described in Section 4.3.1.

Common improvement heuristics for a general MIP are *Relaxation Induced Neighborhood Search* by Danna et al. [7], where the neighborhood between the LP relaxation solution and the current MIP solution is explored, *Local Branching* by Fischetti and Lodi [10], where branching is done in the neighborhood of the current MIP solution, and the general improvement heuristic by Uggen et al. [20], described in Section 4.3.2. A review of general MIP heuristics can be found in Pochet and Wolsey [17].

4.3.1 Constructive heuristic: fix and relax

The concept of the fix and relax decomposition heuristic is to in each iteration divide the integer variables into three groups: one fixed integer block, one integer block, and one continuous block. The fixed integer block consists of fixed solutions found in earlier iterations. The continuous variables are kept continuous throughout the whole solution process.

The planning horizon is split into a finite number of time intervals n. Each time interval makes a subproblem, making the original problem decomposed into n subproblems. The subproblems are solved in iterations corresponding to the time intervals. In the first iteration, the subproblem is solved with the integer variables kept integer in the first time interval, and LP relaxed to continuous variables in the remaining time intervals. The integer variables found for the first interval in the first iteration are kept fixed for the rest of the iterations. In the next iteration, the integer variables found in the first interval are fixed, the integer variables of the second time interval are kept integer, and the integer variables in the remaining time intervals are LP relaxed to continuous variables. This process is repeated until all of the subproblems are solved. The solution found is a complete solution to the original problem. Figure 4.2 illustrates three different iteration stages of the fix and relax heuristic.

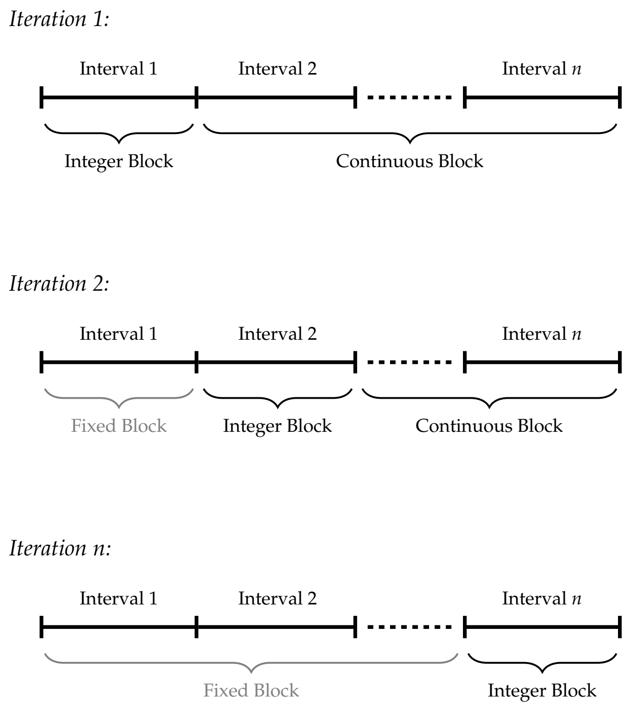


Figure 4.2: Iterations in the fix and relax heuristic

The heuristic was originally described by Dillenberger et al. [8]. Different adjustments of the heuristic have been made. An overview can be found in K.T. Uggen et al. [20]. Time limits and MIP gap limits can be set as stopping criteria in each iteration to speed up the heuristic. The purpose is to obtain a reasonable solution of an iteration in limited time, and then move on to the next iteration. This may, though, lead to infeasibility. Akartunali and Miller [1] suggest that earlier iterations are allocated more time than the later iterations, as the problems are bigger in size and harder to solve at an early stage than later.

4.3.2 Improvement heuristic

The improvement heuristic, based on the "Improvement phase" described in K. T. Uggen et al. [20], takes an already existing feasible solution (for example generated from the fix and relax heuristic), and successively tries to improve it.

The integer variables in the feasible solution are grouped into m groups, split by certain time intervals or split by other criteria. They are all set to be fixed, but in each iteration i, the fixing of the integer variables in the corresponding interval, i, is removed, and the problem is re-optimized. The algorithm loops through the intervals in consecutive order. If a better solution is found during the re-optimization, the integer variable values from this solution are kept as fixed for the future iterations. The algorithm continues to loop until it runs through a full set of intervals, without improvement.

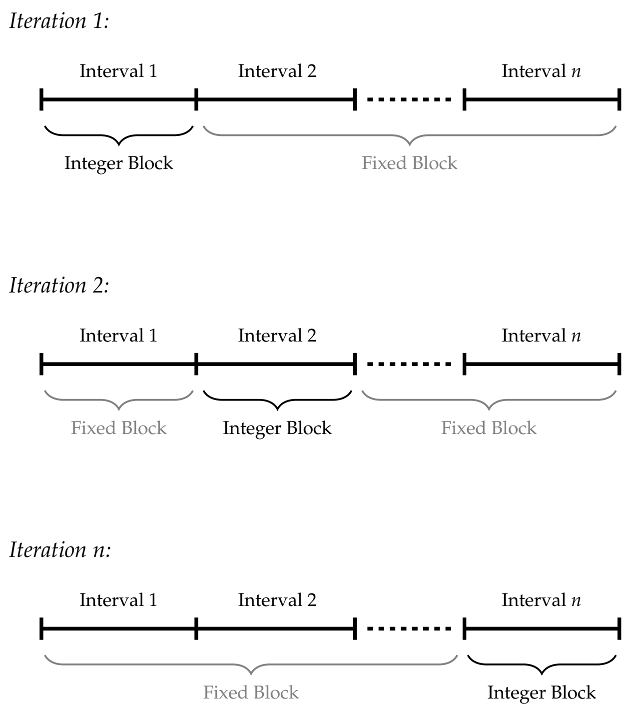


Figure 4.3: Iterations in the improvement heuristic

5. The deterministic model

In this chapter, a deterministic model for the ORSP with elective and emergency demand for surgery is presented and discussed. Section 5.1 presents the mathematical formulation with explanations, and Section 5.2 explains the linearizations done for the overtime costs.

5.1 Formulation of the deterministic model

The following mathematical model is a deterministic model of the problem described in Chapter 2. It is given as a basis for the models incorporating uncertainty, presented in Chapter 6. The uncertainty regarding the arrivals of non-elective patients is not included in the model.

5.1.1 Indices, sets, parameters and variables

Indices

t :	time period index
i :	elective case index
j :	emergency case index
r :	operating room index
d :	duration of stay index
v :	overtime price range index

Sets

$\mathcal{H}$:	set of time periods in the current planning horizon, $\mathcal{H} = \{1, 2, ..., L\}$
$\mathcal{H}_i^C$:	set of time periods in the current planning horizon, and one period after, that elective case i can be scheduled to, $\mathcal{H}_i^C = \{B_i, (B_i + 1), ..., L, (L + 1)\}$
$\mathcal{I}$:	set of elective cases to be scheduled in the planning horizon
$\mathcal{I}^{IC}$:	set of elective cases going to ICU for post-op
$\mathcal{I}^{PC}$:	set of elective cases going to PACU for post-op
$\mathcal{J}_t$:	set of emergency cases in time period t
$\mathcal{J}_t^{IC}$:	set of emergency cases staying at ICU for post-op in time period t
$\mathcal{J}_t^{PC}$:	set of emergency cases staying at PACU for post-op time period t
$\mathcal{R}$:	set of operating rooms
$\mathcal{R}_i^A$:	set of operating rooms that elective case i can go to
$\mathcal{R}_j^B$:	set of operating rooms that emergency case j can go to
$\mathcal{V}$:	set of overtime price ranges

Parameters

L :	last time period in the planning horizon
B_i :	release period, earliest time period for performing elective case i
C^{O}_{trv} :	cost per unit of overtime of price range v in time period t and operating room r
C^{PR}_t :	cost per patient exceeding the capacity of pre-op in time period t
C^{IC}_t :	cost per patient exceeding the capacity of ICU in time period t
C^{PC}_t :	cost per patient exceeding the capacity of PACU in time period t
P^{A}_i :	time needed for performing elective case i
P^{B}_j :	time needed for performing emergency case j
T^{R}_{tr} :	total available regular capacity of operating room r in time period t
T_{trv} :	overtime capacity of price range v in operating room r, and time period t
K^{PR}_t :	available capacity of pre-operative unit in time period t
K^{IC}_t :	available capacity of ICU in time period t
K^{PC}_t :	available capacity of PACU in time period t
D^{IC}_i :	duration of stay at ICU after surgery for elective case i, for $i \in \mathcal{I}^{IC}$, where $D^{IC}_i = 0$ means that the elective case i only stays at ICU in the same time period as the surgery
D^{PC}_i :	duration of stay at PACU after surgery for elective case i, for $i \in \mathcal{I}^{PC}$, where $D^{PC}_i = 0$ means that the elective case i only stays at PACU in the same time period as the surgery

Variables

x_{itr} :	with $x_{itr} = 1$ if elective case i is planned to be performed in time period t and room r. $x_{itr} = 0$ otherwise. $x_{i,(L+1),r} = 1$ implies that elective case i is rejected in the current planning horizon
y_{jr} :	with $y_{jr} = 1$ if emergency case j is performed in room r, $y_{jr} = 0$ otherwise
z_{trv} :	overtime (working time exceeding the total available regular capacity) of price range v, for operating room r, and time period t
e^{PR}_t :	amount of patients exceeding the capacity of pre-op in period t
e^{IC}_t :	amount of patients exceeding the capacity of ICU in period t.
e^{PC}_t :	amount of patients exceeding the capacity of PACU in period t

5.1.2 Mathematical program

$$\min \mathcal{Q} = \sum_{i\in\mathcal{I}} \sum_{t\in\mathcal{H}_i^C} \sum_{r\in\mathcal{R}_i^A} C_{itr} x_{itr} \tag{5.1a}$$

$$+ \sum_{t\in\mathcal{H}} \sum_{r\in\mathcal{R}} \sum_{v\in\mathcal{V}} C^O_{trv} z_{trv} \tag{5.1b}$$

$$+ \sum_{t\in\mathcal{H}} (C_t^{PR} e_t^{PR} + C_t^{IC} e_t^{IC} + C_t^{PC} e_t^{PC}) \tag{5.1c}$$

subject to:

$$\sum_{i\in\mathcal{I}} P_i^A x_{itr} + \sum_{j\in\mathcal{J}_t} P_j^B y_{jr} \le T_{tr}^R + \sum_{v\in\mathcal{V}} z_{trv}, \quad t\in\mathcal{H},\ r\in\mathcal{R} \tag{5.2}$$

$$\sum_{t\in\mathcal{H}_i^C} \sum_{r\in\mathcal{R}_i^A} x_{itr} = 1, \qquad i\in\mathcal{I} \tag{5.3}$$

$$\sum_{r\in\mathcal{R}_j^B} y_{jr} = 1, \qquad t\in\mathcal{H},\ j\in\mathcal{J}_t \tag{5.4}$$

$$\sum_{r\in\mathcal{R}_i^A} \sum_{i\in\mathcal{I}} x_{itr} + |\mathcal{J}_t| \le K_t^{PR} + e_t^{PR}, \qquad t\in\mathcal{H} \tag{5.5}$$

$$\sum_{i\in\mathcal{I}^{IC}} \sum_{\substack{d=\{0,..,D_i^{IC}\}:\\ (t-d)\in\mathcal{H}_i^C}} \sum_{r\in\mathcal{R}_i^A} x_{i(t-d)r} + |\mathcal{J}_t^{IC}| \le K_t^{IC} + e_t^{IC}, \qquad t\in\mathcal{H} \tag{5.6}$$

$$\sum_{i\in\mathcal{I}^{PC}} \sum_{\substack{d=\{0,..,D_i^{PC}\}:\\ (t-d)\in\mathcal{H}_i^C}} \sum_{r\in\mathcal{R}_i^A} x_{i(t-d)r} + |\mathcal{J}_t^{PC}| \le K_t^{PC} + e_t^{PC}, \qquad t\in\mathcal{H} \tag{5.7}$$

$$x_{itr} \in \{0,1\}, \qquad i\in\mathcal{I},\ t\in\mathcal{H}_i^C,\ r\in\mathcal{R}_i^A \tag{5.8}$$

$$y_{jr} \in \{0,1\}, \qquad t\in\mathcal{H},\ j\in\mathcal{J}_t,\ r\in\mathcal{R}_j^B \tag{5.9}$$

$$0 \le z_{trv} \le T_{trv}, \qquad t\in\mathcal{H},\ r\in\mathcal{R},\ v\in\mathcal{V} \tag{5.10}$$

$$e_t^{PR},\ e_t^{IC},\ e_t^{PC} \ge 0, \qquad t\in\mathcal{H} \tag{5.11}$$

5.1.3 Explanation of the model

The purpose of the model is to find the best possible schedule for the elective patient surgeries, depending on a single expected arrival sequence of the non-elective patients. This is done by letting the objective function minimize the total expected costs. The objective function can be split into three parts, (5.1a) - (5.1c):

(5.1a) captures the cost of performing the elective cases in specific rooms and time periods.

(5.1b) calculates the cost of overtime, summing over the different overtime price ranges (linearizations of a curved cost function). Further explanation is given in Section 5.2.

(5.1c) captures the costs of exceeding the capacity in pre-op, ICU and PACU.

The constraints (5.2) regulate the overtime parameters, z_{trv}:

(5.2) calculate the overtime parameters, z_{trv}, in every time period, room, and overtime price range. If the time needed for the elective and emergency cases to be performed in a room and time period exceeds the regular capacity, the corresponding overtime parameters obtain a positive value. The overtime parameters represent overtime of different price ranges, to accommodate for that it may be more expensive to work overtime hours later in the evening than in the early evening. The cost parameters C^{O}_{trv} represent a linearization of a curved cost function, see Figure 5.1. As the costs increase with time, the nature of the objective function will ensure that that the overtime parameters get as small as possible, and that z_{tr1} will obtain a value before z_{tr2}, and so on. The maximum values of the variables are defined in constraints (5.10).

The constraints (5.3) - (5.4) establish certain rules for the variables x_{itr} and y_{jr}:

(5.3) make sure that every elective case gets assigned to one room and one time period. If a case is scheduled to time period $L+1$, the surgery is postponed until the next planning horizon. This can be penalized by giving the cost parameters C_{itr} in the first term of the objective function (5.1a), for $t = L+1$, a large value.

(5.4) make sure that every emergency case gets assigned to a specific room in the time period of arrival.

The constraints (5.5) - (5.7) count if, and by how much, the pre-op and post-op capacities are exceeded. The exceeded capacity variables, e_t^{PR}, e_t^{IC}, and e_t^{PC} are punished with a cost in the objective function (5.1c).

(5.5) measure by how much the capacity of the pre-op is exceeded in the different time periods.

(5.6) measure by how much the capacity of the ICU of the post-op is exceeded in every time period. Patients may stay at post-op for more than a day. These constraints count all patients that are staying in the post-op unit in a time period, also when the surgery is to be performed in an earlier time period. This is done by including the information of the duration parameters when summing over the time indices.

(5.7) do the same as constraints (5.6), but for the PACU.

The constraints **(5.8)** and **(5.9)** define the variables x_{itr} and y_{jr} as binary. The constraints **(5.10)** define the boundaries of the overtime variables, z_{trv}, and the constraints **(5.11)** ensure non-negativity of the variables e_t^{PR}, e_t^{IC}, and e_t^{PC}.

5.2 Linearizations of the overtime costs

The overtime cost parameters represent linearizations of a curved overtime cost function, shown in Figure 5.1. The figure shows an example where the overtime costs are split into 3 price ranges, $v \in \{1, 2, 3\}$. For this example, one time period in the ORSP is defined as 24 hours. Table 5.1 shows how the overtime cost parameters in the example are connected to the hours of a day.

Time		Overtime price range	Parameters	Comments
7 AM to 5 PM	10 hr		$T_{tr}^R = 10$	Time interval of regular capacity
5 PM to 8 PM	3 hr	$v = 1$	$T_{tr1} = 3$	Time interval where the overtime costs C_{tr1}^O per hour
8 PM to 11 PM	3 hr	$v = 2$	$T_{tr2} = 3$	Time interval where the overtime costs C_{tr2}^O per hour
11 PM to 7 AM	8 hr	$v = 3$	$T_{tr3} = 8$	Time interval where the overtime costs C_{tr3}^O per hour

Table 5.1: Overtime cost ranges - example

As the overtime prices increase during the evening and night, the overtime cost parameters are related to each other in this way: $C_{tr1}^O \leq C_{tr2}^O \leq C_{tr3}^O$. The nature of the objective function will ensure that the overtime variables of the first price

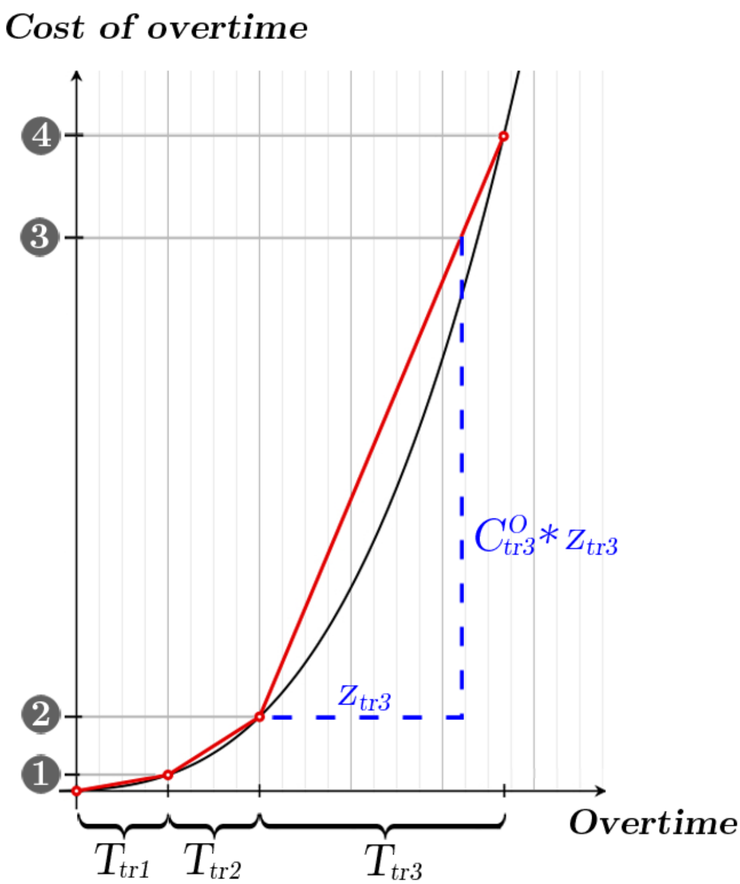


Figure 5.1: Linearizations of overtime costs - example

range will be used before the ones of the second price range, and so on.

The total overtime costs for the example will be as follows:

$$\sum_{t\in\mathcal{H}}\sum_{r\in\mathcal{R}}\sum_{v\in\mathcal{V}} C^O_{trv} z_{trv} = \sum_{t\in\mathcal{H}}\sum_{r\in\mathcal{R}} (C^O_{tr1} z_{tr1} + C^O_{tr2} z_{tr2} + C^O_{tr3} z_{tr3})$$

Depending on the total overtime, either only the first term will obtain a value, or the first and the second term, or all three of the terms.

On the vertical axis of Figure 5.1, four overtime cost points for a given time period t and room r are marked:

Point 1 marks the total overtime cost for 3 hours of overtime, $T_{tr1} = 3$

Total overtime cost = $C^O_{tr1} T_{tr1}$.

$z_{tr1} = T_{tr1}$

Point 2 marks the total overtime cost for 6 hours of overtime, $(T_{tr1} + T_{tr2}) = 6$

Total overtime cost = $(C^O_{tr1} T_{tr1} + C^O_{tr2} T_{tr2})$.

$z_{tr1} = T_{tr1}, \ z_{tr2} = T_{tr2}$

Point 3 marks the total overtime cost for a value between 6 and 14 hours of overtime, $(T_{tr1} + T_{tr2} + z_{tr3}) = (6 + z_{tr3})$

Total overtime cost = $(C^O_{tr1} T_{tr1} + C^O_{tr2} T_{tr2} + C^O_{tr3} z_{tr3})$

$z_{tr1} = T_{tr1}, \quad z_{tr2} = T_{tr2}, \quad 0 \leq z_{tr3} \leq T_{tr3}$

Point 4 marks the total overtime cost for 14 hours of overtime, $(T_{tr1} + T_{tr2} + T_{tr3})$ $= 14$

Total overtime cost = $(C^O_{tr1} T_{tr1} + C^O_{tr2} T_{tr2} + C^O_{tr3} T_{tr3})$

$z_{tr1} = T_{tr1}, \quad z_{tr2} = T_{tr2}, \quad z_{tr3} = T_{tr3}$

6. The stochastic multi-stage model

To incorporate the uncertainty regarding the arrivals of non-elective patients, the deterministic model described in Chapter 5 is reformulated to a stochastic multi-stage model. The purpose of this model is, as in the deterministic, to make a best possible schedule of the elective cases in the planning period, creating a robust solution for both the elective and non-elective patients, no matter what scenario that actually will occur.

The model solves the problem, regarding the possibilities of the different scenarios. Each scenario represents a certain distribution of emergency patients arriving throughout the planning horizon, and is given a likely probability of occurring. Many of the input parameters may change with each scenario.

Section 6.1 presents the recourse decisions to be made in every stage of the model. Section 6.2 presents the scenario formulation of the multi-stage model. The decision making sequence is explained in Section 6.3, and the node formulation of the multi-stage model is given in Section 6.4. Alterations of the model is given in Section 6.5. Last, in Section 6.6, a small example case is presented.

6.1 Recourse decisions

In every stage of the model, the decisions of allocating the emergency patients to ORs have to be made. In addition, two other types of recourse decisions are allowed to be made in every stage. The decisions are made for the elective patients that are scheduled to be performed in the time period of the stage, and they are made to fit in the emergency patients that arrive in a best possible way.

Recourse decision 1: Decide if some of the scheduled elective patients should be postponed until the next time period.

Recourse decision 2: Decide if some of the scheduled elective patients should change OR within the same time period.

6.2 The scenario formulation of the stochastic model

In the scenario formulation each possible scenario for the problem is formulated as a deterministic problem, and non-anticipativity constraints are added to ensure that the information structure associated with the decision process is honored, as described in Section 6.3. This leads to a large amount of variables and constraints, but it is an intuitive model, making it quite straightforward to read and understand the problem.

6.2.1 Indices, sets, parameters and variables

Indices

s :	scenario index
t :	time period index
i :	elective case index
j :	emergency case index
r :	operating room index
d :	duration of stay index
v :	overtime price range index
n :	node index

Sets

$\mathcal{S}$:	set of scenarios denoting the possible realizations of emergency patient arrivals
$\mathcal{H}$:	set of time periods in the current planning horizon, $\mathcal{H} = \{1, 2, ..., L\}$
$\mathcal{H}_i^C$:	set of time periods in the current planning horizon, and one period after, that elective case i can be scheduled to, $\mathcal{H}_i^C = \{B_i, (B_i + 1), ..., L, (L + 1)\}$
$\mathcal{H}_i^D$:	$\mathcal{H}_i^D = \{B_i, (B_i + 1), ..., L\}$
$\mathcal{H}_i^F$:	$\mathcal{H}_i^F = \{(B_i + 1), (B_i + 2), ..., L\}$
$\mathcal{I}$:	set of elective cases to be scheduled in the planning horizon
$\mathcal{I}_t^B$:	set of elective cases with release period B_i in time period t
$\mathcal{I}^{IC}$:	set of elective cases going to ICU for post-op
$\mathcal{I}^{PC}$:	set of elective cases going to PACU for post-op
$\mathcal{J}_{ts}$:	set of emergency cases in time period t and scenario s
$\mathcal{J}_{ts}^{IC}$:	set of emergency cases staying at ICU for post-op in time period t and scenario s
$\mathcal{J}_{ts}^{PC}$:	set of emergency cases staying at PACU for post-op time period t and scenario s

$\mathcal{R}$:	set of operating rooms
$\mathcal{R}_i^A$:	set of operating rooms that elective case i can go to
$\mathcal{R}_{js}^B$:	set of operating rooms that emergency case j in scenario s can go to
$\mathcal{N}^H$:	set of nodes in the time periods of the planning horizon, $t(n) \in \mathcal{H}$, i.e. all nodes in the scenario tree, except from the origin node, $n = 1$, and the leaf nodes (nodes in the time period $t(n) = L + 1$)
$\mathcal{S}_n^N$:	set of scenarios that pass through node n in the scenario tree
$\mathcal{V}$:	set of overtime price ranges

Parameters

w_s :	probability of scenario s occurring, $\sum_{s \in \mathcal{S}} w_s = 1$
$t(n)$:	The time period corresponding to node n
L :	last time period in the planning horizon
B_i :	release period, earliest time period for performing elective case i
C_{itrs}^A :	cost of performing elective case i in time period t, operating room r, and scenario s, for $t \in \mathcal{H}_i^C$
C_{itr}^{R1} :	cost of postponing elective case i one time period, to time period $t+1$, and operating room r, for $t \in \mathcal{H}_i^D$. Comes in addition to C_{itrs}^A
C_{itr}^{R2} :	cost of moving elective case i in time period t from operating room r' to operating room r, $r \neq r'$, for $t \in \mathcal{H}_i^D$. Comes in addition to C_{itrs}^A
C_{trv}^O :	cost per unit of overtime of price range v in time period t and operating room r
C_t^{PR} :	cost per patient exceeding the capacity of pre-op in time period t
C_t^{IC} :	cost per patient exceeding the capacity of ICU in time period t
C_t^{PC} :	cost per patient exceeding the capacity of PACU in time period t
P_{is}^A :	time needed for performing elective case i in scenario s
P_{js}^B :	time needed for performing emergency case j in scenario s
T_{trs}^R :	total available regular capacity of operating room r in time period t in scenario s
T_{trsv} :	overtime capacity of price range v in operating room r, time period t, and scenario s
K_t^{PR} :	available capacity of pre-operative unit in time period t
K_t^{IC} :	available capacity of ICU in time period t
K_t^{PC} :	available capacity of PACU in time period t

D_{is}^{IC} : duration of stay at ICU after surgery for elective case i, scenario s, for $i \in \mathcal{I}^{IC}$, where $D_{is}^{IC} = 0$ means that the elective case i only stays at ICU in the same time period as the surgery

D_{is}^{PC} : duration of stay at PACU after surgery for elective case i, scenario s, for $i \in \mathcal{I}^{PC}$, where $D_{is}^{PC} = 0$ means that the elective case i only stays at PACU in the same time period as the surgery

Variables

x_{itr} : with $x_{itr} = 1$ if elective case i is planned to be performed in time period t and room r. $x_{itr} = 0$ otherwise. $x_{i,(L+1),r} = 1$ implies that elective case i is rejected in the current planning horizon

y_{jrs} : with $y_{jrs} = 1$ if emergency case j is performed in room r and scenario s, $y_{jrs} = 0$ otherwise

z_{trsv} : overtime (working time exceeding the total available regular capacity) of price range v, for time period t, operating room r, and scenario s

e_{ts}^{PR} : amount of patients exceeding the capacity of pre-op in period t and scenario s

e_{ts}^{IC} : amount of patients exceeding the capacity of ICU in period t and scenario s

e_{ts}^{PC} : amount of patients exceeding the capacity of PACU in period t and scenario s

x_{itrs}^{R1} : recourse variable regarding postponement of elective patients, to be decided in time period t. $x_{itrs}^{R1} = 1$ if elective case i is postponed one time period, from time period t to time period $t+1$ and room r, in scenario s ($x_{itr} = 1$). $x_{itrs}^{R1} = 0$ otherwise

x_{itrs}^{R2} : recourse variable regarding change of rooms for elective patients, to be decided in time period t. $x_{itr's}^{R2} = 1$ if elective case i in time period t and scenario s is moved from operating room r to operating room r', $r \neq r'$ and $x_{itr} = 1$. $x_{itrs}^{R2} = 0$ otherwise

x_{itrs}^{A} : with $x_{itrs}^{A} = 1$ if elective case i *actually* is performed in time period t, room r, and scenario s. $x_{itrs}^{A} = 0$ otherwise

x_{irn}^{N} : variable to ensure non-anticipativity of the recourse variables decided in node n.

y_{jrn}^{N} : variable to ensure non-anticipativity of the emergency patient arrivals knowledge in node n

6.2.2 Mathematical program

$$min\ \mathcal{Q}^{\mathcal{S}} = \sum_{s\in\mathcal{S}} w_s \Big(\tag{6.1a}$$

$$\sum_{i\in\mathcal{I}} \sum_{t\in\mathcal{H}_i^C} \sum_{r\in\mathcal{R}_i^A} C_{itrs}^{A} x_{itrs}^{A} \tag{6.1b}$$

$$+\sum_{i\in\mathcal{I}} \sum_{t\in H_i^D} \sum_{r\in\mathcal{R}_i^A} C_{itr}^{R1} x_{itrs}^{R1} \tag{6.1c}$$

$$+\sum_{i\in\mathcal{I}} \sum_{t\in H_i^D} \sum_{r\in\mathcal{R}_i^A} C_{itr}^{R2} x_{itrs}^{R2} \tag{6.1d}$$

$$+\sum_{t\in\mathcal{H}} \sum_{r\in\mathcal{R}} \sum_{v\in\mathcal{V}} C_{trv}^{O} z_{trsv} \tag{6.1e}$$

$$+\sum_{t\in\mathcal{H}} (C_t^{PR} e_{ts}^{PR} + C_t^{IC} e_{ts}^{IC} + C_t^{PC} e_{ts}^{PC}) \Big) \tag{6.1f}$$

subject to:

$$x_{itr} + x^{R2}_{itrs} - \sum_{r' \in \mathcal{R}^A_i} x^{R1}_{itr's} - \sum_{r' \in \mathcal{R}^A_i \setminus \{r\}} x^{R2}_{itr's} \leq x^A_{itrs} \qquad i \in \mathcal{I},\ t = B_i,\ r \in \mathcal{R}^A_i,\ s \in \mathcal{S} \tag{6.2}$$

$$x_{itr} + x^{R1}_{i,(t-1),rs} + x^{R2}_{itrs} - \sum_{r' \in \mathcal{R}^A_i} x^{R1}_{itr's} - \sum_{r' \in \mathcal{R}^A_i \setminus \{r\}} x^{R2}_{itr's} \leq x^A_{itrs} \qquad i \in \mathcal{I},\ t \in \mathcal{H}^F_i,\ r \in \mathcal{R}^A_i,\ s \in \mathcal{S} \tag{6.3}$$

$$x_{itr} + x^{R1}_{i,(t-1),rs} \leq x^A_{itrs} \qquad i \in \mathcal{I},\ t = (L+1),\ r \in \mathcal{R}^A_i,\ s \in \mathcal{S} \tag{6.4}$$

$$\sum_{i \in \mathcal{I}} P^A_{is} x^A_{itrs} + \sum_{j \in \mathcal{J}_{ts}} P^B_{js} y_{jrs} \leq T^R_{trs} + \sum_{v \in \mathcal{V}} z_{trsv}, \qquad t \in \mathcal{H},\ r \in \mathcal{R},\ s \in \mathcal{S} \tag{6.5}$$

$$\sum_{t \in \mathcal{H}^C_i} \sum_{r \in \mathcal{R}^A_i} x_{itr} = 1, \qquad i \in \mathcal{I} \tag{6.6}$$

$$\sum_{r \in \mathcal{R}^B_{js}} y_{jrs} = 1, \qquad t \in \mathcal{H},\ s \in \mathcal{S},\ j \in \mathcal{J}_{ts} \tag{6.7}$$

$$\sum_{r \in \mathcal{R}^A_i} \left(x_{itr} - x^{R1}_{itrs} - x^{R2}_{itrs} \right) \geq 0 \qquad i \in \mathcal{I},\ t \in \mathcal{H}^D_i,\ s \in \mathcal{S} \tag{6.8}$$

$$\sum_{r \in \mathcal{R}^A_i} \Big(\sum_{i \in \mathcal{I}^B_t} x_{itr} + \sum_{i \in \mathcal{I} \setminus \mathcal{I}^B_t} (x_{itr} + x^{R1}_{i,(t-1),rs}) \Big) + |\mathcal{J}_{ts}| \leq K^{PR}_t + e^{PR}_{ts}, \qquad t \in \mathcal{H},\ s \in \mathcal{S} \tag{6.9}$$

$$\sum_{i \in \mathcal{I}^{IC}} \sum_{\substack{d = \{0,..,D^{IC}_{is}\}: \\ (t-d) \in \mathcal{H}^C_i}} \sum_{r \in \mathcal{R}^A_i} x^A_{i,(t-d),rs} + |\mathcal{J}^{IC}_{ts}| \leq K^{IC}_t + e^{IC}_{ts}, \qquad t \in \mathcal{H},\ s \in \mathcal{S} \tag{6.10}$$

$$\sum_{i \in \mathcal{I}^{PC}} \sum_{\substack{d = \{0,..,D^{PC}_{is}\}: \\ (t-d) \in \mathcal{H}^C_i}} \sum_{r \in \mathcal{R}^A_i} x^A_{i,(t-d),rs} + |\mathcal{J}^{PC}_{ts}| \leq K^{PC}_t + e^{PC}_{ts}, \qquad t \in \mathcal{H},\ s \in \mathcal{S} \tag{6.11}$$

$$x^{R1}_{i,t(n),rs} = x^{N1}_{irn}, \qquad i \in \mathcal{I},\ n \in \mathcal{N}^H,\ r \in \mathcal{R}^A_i,\ s \in \mathcal{S}^N_n \tag{6.12}$$

$$x^{R2}_{i,t(n),rs} = x^{N2}_{irn}, \qquad i \in \mathcal{I},\ n \in \mathcal{N}^H,\ r \in \mathcal{R}^A_i,\ s \in \mathcal{S}^N_n \tag{6.13}$$

$$y_{jrs} = y^N_{jrn}, \qquad n \in \mathcal{N}^H,\ s \in \mathcal{S}^N_n,\ j \in \mathcal{J}_{t(n),s},\ r \in \mathcal{R}^B_{js} \tag{6.14}$$

$$x_{itr} \in \{0,1\}, \qquad i \in \mathcal{I},\ t \in \mathcal{H}^C_i,\ r \in \mathcal{R}^A_i \tag{6.15}$$

$$y_{jrs} \in \{0,1\}, \qquad t \in \mathcal{H},\ s \in \mathcal{S},\ j \in \mathcal{J}_{ts},\ r \in \mathcal{R}^B_{js} \tag{6.16}$$

$$x^N_{irn} \in \{0,1\}, \qquad i \in \mathcal{I},\ r \in \mathcal{R}^A_i,\ n \in \mathcal{N}^H \tag{6.17}$$

$$y^N_{jrn} \in \{0,1\}, \qquad n \in \mathcal{N}^H,\ s \in \mathcal{S}^N_n,\ j \in \mathcal{J}_{t(n),s},\ r \in \mathcal{R}^B_{js} \tag{6.18}$$

$$x^{R1}_{itrs} \in \{0,1\}, \qquad i \in \mathcal{I},\ t \in \mathcal{H}^D_i,\ r \in \mathcal{R}^A_i,\ s \in \mathcal{S} \tag{6.19}$$

$$x^{R2}_{itrs} \in \{0,1\}, \qquad i \in \mathcal{I},\ t \in \mathcal{H}^D_i,\ r \in \mathcal{R}^A_i,\ s \in \mathcal{S} \tag{6.20}$$

$$x^{A}_{itrs} \in \{0,1\}, \qquad i \in \mathcal{I},\ t \in \mathcal{H}^C_i,\ r \in \mathcal{R}^A_i,\ s \in \mathcal{S} \tag{6.21}$$

$$0 \leq z_{trsv} \leq T_{trsv}, \qquad t \in \mathcal{H},\ r \in \mathcal{R},\ s \in \mathcal{S},\ v \in \mathcal{V} \tag{6.22}$$

$$e^{PR}_{ts},\ e^{IC}_{ts},\ e^{PC}_{ts} \geq 0, \qquad t \in \mathcal{H},\ s \in \mathcal{S} \tag{6.23}$$

6.2.3 Explanation of the model

The purpose of the model is to find the best possible schedule for the elective patient surgeries, taking into account the uncertainty regarding the emergency patient arrivals. This is done by letting the objective function minimize the total expected costs. The objective function can be split into six parts, (6.1a) - (6.1f):

(6.1a) weights the total costs of each possible scenario, depending on the probabilities for them occurring.

(6.1b) captures the cost of performing the elective cases in specific rooms and time periods. The variables x^A_{itrs} represent when and where the elective cases actually are performed. They will, per time period, include the elective patients that are postponed from the last time period, and exclude the elective patients that are postponed until the next. They will also include any changes of operating rooms.

(6.1c) captures the cost of *postponing* the elective cases to specific rooms and time periods. If a recourse variable, x^{R1}_{itrs}, takes value 1 for a given patient, the surgery will be postponed one time period. Any extra costs due to the postponement are included in this term.

(6.1d) captures the cost of *changing the operating room* of the elective cases in specific time periods. If a recourse variable, $x^{R2}_{itr's}$, takes value 1 for a given patient, the surgery will be moved within the same time period from room r to room r' ($r \neq r'$). Any extra costs due to the change of operating room are included in this term.

(6.1e) calculates the cost of overtime, summing over the different overtime price ranges (linearizations of a curved cost function). A more detailed description is given in Section 5.2.

(6.1f) captures the costs of exceeding the capacity in pre-op, ICU and PACU.

The constraints (6.2) - (6.4) calculate the variables carrying information about where and when the elective cases *actually* are performed, x^A_{itrs}:

(6.2) calculate the x^A_{itrs} for the elective cases being treated in their release periods. The recourse variables, x^{R2}_{itrs}, are added to the originally scheduled, to include the patients that have changed room. x^{R1}_{itrs} and the rest of the x^{R2}_{itrs}, are subtracted from the originally scheduled, x_{itr}, to include that the elective patients may be postponed one day or changed operating room for. The summations over R^A_i are done to subtract the recourse variables no matter which room the elective patients may be postponed or changed to. This can be done, as the x^A_{itrs} in constraints (6.21) are declared as binary variables, making it impossible for them to obtain a negative value.

(6.3) calculate the x^{A}_{itrs} for the elective cases planned to be performed in the time periods after their release periods, within the planning horizon. The elective patients that are postponed from the preceding time period, $x^{R1}_{i,(t-1),rs}$, and the elective patients with changed operating rooms, x^{R2}_{itrs}, are *added* to the originally scheduled patients, x_{itrs} for each elective patient, time period, and room. The elective patients that are being postponed until the next time period, x^{R1}_{itrs}, and the elective patients with changed operating rooms within the time period, x^{R2}_{itrs}, are *subtracted* in the same way as in constraints (6.2).

(6.4) calculate the x^{A}_{itrs} for the last time period, $t = L + 1$. If $x^{A}_{itrs} = 1$ for any elective patients in this time period, it means that they are not being treated until the next planning horizon. The elective patients that are postponed from the last time period in the horizon, $x^{R1}_{i,(t-1),rs}$, $(t-1) = L$, are added to the patients originally scheduled to the next planning horizon, x_{itr}, $t = L + 1$.

The constraints (6.5) regulate the overtime parameters, z_{trsv}:

(6.5) calculate the overtime parameters, z_{trsv}, in every time period, room, scenario and overtime price range. If the time needed for the elective and emergency cases to be performed in a room and time period of a scenario exceeds the regular capacity, the corresponding overtime parameters obtain a positive value. The overtime parameters represent overtime of different price ranges, to accommodate for that it may be more expensive to work overtime hours later in the evening than in the early evening. The cost parameters C^{O}_{trv} represent linearizations of a curved cost function, see Figure 5.1. As the costs increase with time, the nature of the objective function will ensure that that the overtime parameters get as small as possible, and that z_{trs1} will obtain a value before z_{trs2}, and so on. The maximum values of the variables are defined in constraints (6.22).

The constraints (6.6) - (6.8) establish certain rules for the variables x_{itr}, y_{jrs} and the recourse variables, x^{R1}_{itrs} and x^{R2}_{itrs}.

(6.6) make sure that every elective case gets assigned to one room and one time period in the original plan. If a case is scheduled to time period $L + 1$, the surgery is postponed until the next planning horizon. This can be penalized by giving the cost parameters C^{A}_{itrs} in the first term of the objective function (6.1a), for $t = L + 1$, a large value.

(6.7) make sure that every emergency case gets assigned to a specific room in the time period of arrival. If it is not planned for enough capacity, the arrivals of the emergency patients may force some of the originally scheduled elective patients to be postponed or given another operating room.

(6.8) make sure that the recourse decisions only can be made for the elective patients that originally are scheduled to the time period that the decision has to be made. The two recourse variables, x^{R1}_{itrs} and x^{R2}_{itrs}, can only obtain value 1 for a patient if the originally scheduled x_{itr} have value 1. Only one of the recourse variables can obtain value 1 at the same time (an elective patient can not both be given a new room in the time period of treatment and be postponed at the same time). The constraints also make sure that the elective patients only may be postponed *one* day, and that an already postponed patient can not be postponed further. By summing over the set of possible rooms, we also make sure that a patient may change OR if postponed.

The constraints (6.9) - (6.11) count if, and by how much, the pre-op and post-op capacities are exceeded. The exceeded capacity variables, e^{PR}_{ts}, e^{IC}_{ts}, and e^{PC}_{ts} are punished with a cost in the objective function (6.1c).

(6.9) measure by how much the capacity of the pre-op is exceeded in the different time periods and scenarios. The exceeded capacity variables are punished with a cost in the objective function (6.1f). The first term regards the elective patients that originally are scheduled to their release period. They will be at the pre-op no matter what recourse decisions are being made. In the next term, for the rest of the elective patients, the recourse decisions regarding postponement are included. The elective patients that are postponed from the preceding time period are added ($x^{R1}_{i,(t-1),rs}$). The elective patients that are being postponed to the next time period are also included - as they would have to spend an extra night at the hospital.

(6.10) measure by how much the capacity of the ICU of the post-op is exceeded in every time period and every scenario. Patients may stay at post-op for more than a day. These constraints count all patients that are staying in the post-op unit in a time period, also when the surgery is to be performed in an earlier time period. This is done by including the information of the duration parameters when summing over the time indices.

(6.11) do the same as constraints (6.10), but for the PACU.

To honor the non-anticipativity of the decision process, the constraints (6.12) - (6.14) are added.

(6.12) ensure that the decision of which elective patients to postpone only depends on the information available in the period they are being postponed *from*. This decision will count no matter how many emergency patients arrive the time period they are being postponed to. This is done by introducing a variable for each node within the planning horizon, x^{N1}_{irn}, $t(n) \in \mathcal{H}$, set to be equal to the recourse decision taken in this specific node.

(6.13) ensure that the decision of which elective patients to change room for, in the same time period, only depends on the information available in the time period they are being treated in. This is done by introducing a variable for each node, x_{irn}^{N2}, $t(n) \in \mathcal{H}$, set to be equal to the recourse decision taken in this specific node.

(6.14) ensure that, for all scenarios going through a specific node, the amount of emergency patients arriving in the time period of the node and the allocation of them will be the same. This is done by introducing a variable for each node, y_{jrn}^{N}, $t(n) \in \mathcal{H}$, making the relevant y_{jrs} equal.

The constraints **(6.15)** - **(6.21)** define the variables x_{itr}, y_{jrs}, x_{irn}^{N1}, x_{irn}^{N2}, y_{jrn}^{N}, x_{itrs}^{R1}, x_{itrs}^{R2}, and x_{itrs}^{A} as binary. The constraints **(6.22)** define the boundaries of the overtime variables, z_{trsv}, and the constraints **(6.23)** ensure non-negativity of the variables e_{ts}^{PR}, e_{ts}^{IC}, and e_{ts}^{PC}.

6.3 Decision making sequence

The decision making sequence is illustrated by a small example. The planning period of the example consists of three days: Monday to Wednesday. It is possible to postpone the elective patients until after Wednesday by scheduling them to time period $L+1$. These patients are not being treated in the current planning horizon.

In each time period of the example, the amount of emergency patients that arrive may be *high* or *low*. With a horizon of three days, 8 scenarios are possible in total. This can be seen as 8 end-leaf nodes in Figure 6.1. The initial decisions of when to treat the elective patients are taken on Sunday evening. During Monday, emergency patients arrive and are given rooms to be treated in. The recourse decisions that must be taken during the day (illustrated as *Monday evening*) is whether some of the elective patients, originally scheduled to Monday, should be postponed one day, or if they should be given a new OR, or be performed as planned. An overview of the decision making sequence for the example is given in Table 6.1.

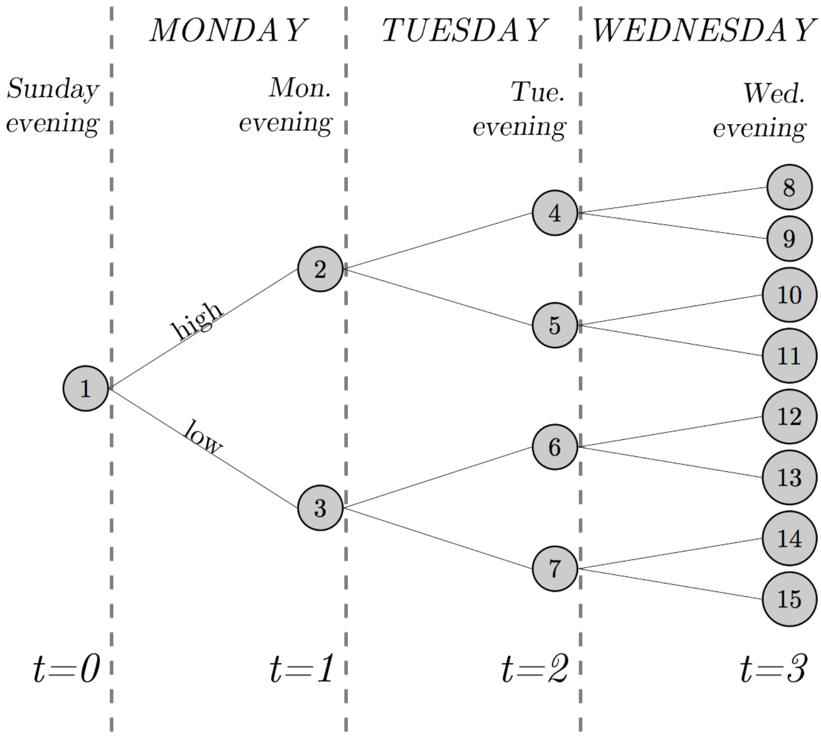


Figure 6.1: Scenario tree for the example illustrating the decision making sequence

Day	$t=$	Variables decided		Description
Sun. eve.	0	x_{itr},	$i \in \mathcal{I},\ t \in \mathcal{H}_i^C,\ r \in \mathcal{R}_i^A$	Original schedule of when the elective patients are to be treated is decided.
Mon.	1	y_{jrs},	$t = 1,\ s \in \mathcal{S},\ j \in \mathcal{J}_{ts},\ r \in \mathcal{R}_j^B$	The emergency patients that arrive on Mon. (either a high or low amount) have to be treated, and are given specific ORs.
		x_{itrs}^{R1},	$t = 1,\ i \in \mathcal{I},\ r \in \mathcal{R}_i^A,\ s \in \mathcal{S}$	Some of the elective patients, originally scheduled to Mon. are being postponed until Tue. $(t+1)$
		x_{itrs}^{R2},	$t = 1,\ i \in \mathcal{I},\ r \in \mathcal{R}_i^A,\ s \in \mathcal{S}$	Some of the elective patients are moved from one OR to another, but are still treated on Mon.
Tue.	2	y_{jrs},	$t = 2,\ s \in \mathcal{S},\ j \in \mathcal{J}_{ts},\ r \in \mathcal{R}_j^B$	The emergency patients that arrive on Tue. (either a high or low amount) have to be treated, and are given specific ORs.
		x_{itrs}^{R1},	$t = 2,\ i \in \mathcal{I},\ r \in \mathcal{R}_i^A,\ s \in \mathcal{S}$	Some of the elective patients, originally scheduled to Tue. are being postponed until Wed. $(t+1)$
		x_{itrs}^{R2},	$t = 2,\ i \in \mathcal{I},\ r \in \mathcal{R}_i^A,\ s \in \mathcal{S}$	Some of the elective patients are moved from one OR to another, but are still treated on Tue.
Wed.	3	y_{jrs},	$t = 3,\ s \in \mathcal{S},\ j \in \mathcal{J}_{ts},\ r \in \mathcal{R}_j^B$	The emergency patients that arrive on Wed. (either a high or low amount) have to be treated, and are given specific ORs.
		x_{itrs}^{R1},	$t = 3,\ i \in \mathcal{I},\ r \in \mathcal{R}_i^A,\ s \in \mathcal{S}$	Some of the elective patients, originally scheduled to Wed. are being postponed until the next planning period. $(t = L+1)$
		x_{itrs}^{R2},	$t = 3,\ i \in \mathcal{I},\ r \in \mathcal{R}_i^A,\ s \in \mathcal{S}$	Some of the elective patients are moved from one OR to another, but are still treated on Wed.

Table 6.1: Decision making sequence for the example

6.4 The node formulation of the stochastic model

To reduce the number of variables and constraints, the scenario formulation of the stochastic model is rewritten to a node formulation. Each node represents a specific point in time, and contains information about the entire history until that time period.

6.4.1 Indices, sets, parameters and variables

Indices

t :	time period index
i :	elective case index
j :	emergency case index
r :	operating room index
d :	duration of stay index
n :	node index
v :	overtime price range index

Sets

$\mathcal{N}$:	set of nodes in the scenario tree, indexed by n
$\mathcal{N}_t$:	set of nodes in time period $t = t(n)$
$\mathcal{H}$:	set of time periods in the current planning horizon, $\mathcal{H} = \{1, 2, ..., L\}$
$\mathcal{H}_i^C$:	$\mathcal{H}_i^D = \{B_i, (B_i + 1), ..., L, (L + 1)\}$
$\mathcal{I}$:	set of elective cases to be scheduled in the planning horizon
$\mathcal{I}_n^B$:	set of elective cases with release period B_i in the time period of the node n
$\mathcal{I}^{IC}$:	set of elective cases going to ICU for post-op
$\mathcal{I}^{PC}$:	set of elective cases going to PACU for post-op
$\mathcal{J}_n$:	set of emergency cases, arriving in the time period $t(n)$, for the scenarios passing through node n
$\mathcal{J}_n^{IC}$:	set of emergency cases staying at ICU for post-op in the time period $t(n)$ for the scenarios passing through node n
$\mathcal{J}_n^{PC}$:	set of emergency cases staying at PACU for post-op in the time period $t(n)$ for the scenarios passing through node n
$\mathcal{R}$:	set of operating rooms
$\mathcal{R}_i^A$:	set of operating rooms that elective case i can go to

$\mathcal{R}^B_{jn}$: set of operating rooms that emergency case j arriving in the time period $t(n)$, in the scenarios passing through node n, can go to

$\mathcal{N}^C_i$: set of nodes in the time periods in the current planning horizon, and one period after, that elective case i can be scheduled to: $\{B_i, (B_i + 1), ..., L, (L + 1)\}$

$\mathcal{N}^D_i$: set of nodes in the time periods: $\{B_i, (B_i + 1), ..., L\}$

$\mathcal{N}^F_i$: set of nodes in the time periods: $\{(B_i + 1), (B_i + 2), ..., L\}$

$\mathcal{N}^H$: set of nodes in the time periods of the planning horizon, $t(n) \in \mathcal{H}$, i.e. all nodes the scenario tree, except from the origin node, $n = 1$, and the leaf nodes (nodes in the time period $t(n) = L + 1$)

$\mathcal{N}^P_n$: set of nodes: node n and the nodes preceding node n in the scenario tree

$\mathcal{V}$: set of overtime price ranges

Parameters

m_n : probability of visiting node n

$p(n)$: preceding node of node n

$t(n)$: time period corresponding to node n

L : last time period in the planning horizon

B_i : release period, earliest time period for performing elective case i

C^A_{irn} : cost of performing elective case i in operating room r and time period $t(n)$ for the scenarios passing through node n, for $n \in \mathcal{N}^C_i$

C^{R1}_{itr} : cost of postponing elective case i one time period, to time period $t + 1$, and operating room r, for $t \in \mathcal{H}^D_i$. Comes in addition to C^A_{irn}

C^{R2}_{itr} : cost of moving elective case i in time period t from operating room r' to operating room r, $r \neq r'$, for $t \in \mathcal{H}^D_i$. Comes in addition to C^A_{irn}

C^O_{trv} : cost per unit of overtime of price range v in time period t and operating room r

C^{PR}_t : cost per patient exceeding the capacity of pre-op in time period t

C^{IC}_t : cost per patient exceeding the capacity of ICU in time period t

C^{PC}_t : cost per patient exceeding the capacity of PACU in time period t

P^A_{in} : time needed for performing elective case i in the scenarios passing through node n

P^B_{jn} : time needed for performing emergency case j in the scenarios passing through node n

T^R_{rn} : total available regular capacity of operating room r in time period $t(n)$ in the scenarios passing through node n

T_{rnv} : overtime capacity of price range v in operating room r and time period $t(n)$ in the scenarios passing through node n

K_t^{PR} : available capacity of pre-operative unit in time period t
K_t^{IC} : available capacity of ICU in time period t
K_t^{PC} : available capacity of PACU in time period t
D_{in}^{IC} : duration of stay at ICU after surgery for elective case i, in the scenarios passing through node n, for $i \in \mathcal{I}^{IC}$, where $D_{in}^{IC} = 0$ means that the elective case i only stays at ICU in the same time period as the surgery
D_{in}^{PC} : duration of stay at PACU after surgery for elective case i, in the scenarios passing through node n, for $i \in \mathcal{I}^{PC}$, where $D_{in}^{PC} = 0$ means that the elective case i only stays at PACU in the same time period as the surgery

Variables

x_{itr} : with $x_{itr} = 1$ if elective case i is planned to be performed in time period t and room r. $x_{itr} = 0$ otherwise. $x_{i(L+1)r} = 1$ implies that elective case i is rejected in the current planning horizon

y_{jrn} : with $y_{jrn} = 1$ if emergency case j, arriving in time period $t(n)$, is performed in room r for the scenarios passing through node n. $y_{jrn} = 0$ otherwise

z_{rnv} : overtime (working time exceeding the total available regular capacity) of price range v, for OR r, time period $t(n)$ and the scenarios passing through node n

e_n^{PR} : amount of patients exceeding the capacity of pre-op in period $t(n)$ and the scenarios passing through node n

e_n^{IC} : amount of patients exceeding the capacity of ICU in period $t(n)$ and the scenarios passing through node n

e_n^{PC} : amount of patients exceeding the capacity of PACU in period $t(n)$ and the scenarios passing through node n

x_{irn}^{R1} : recourse variable regarding postponement of elective patients, to be decided in time period $t(n)$. $x_{irn}^{R1} = 1$ if elective case i is postponed one time period ($x_{irn} = 1$), from time period $t(n)$ to time period $t(n) + 1$ and room r, in the scenarios passing through node n. $x_{irn}^{R1} = 0$ otherwise

x_{irn}^{R2} : recourse variable regarding change of rooms for elective patients, to be decided in time period $t(n)$. $x_{ir'n}^{R2} = 1$ if elective case i in time period $t(n)$ and the scenarios passing through node n is moved from operating room r to operating room r', $r \neq r'$, and $x_{irn} = 1$. $x_{irn}^{R2} = 0$ otherwise

x_{irn}^{A} : with $x_{irn}^{A} = 1$ if elective case i *actually* is performed in time period $t(n)$, room r, and the scenarios passing through node n. $x_{irn}^{A} = 0$ otherwise

6.4.2 Mathematical program

$$\begin{aligned}
min\ \mathcal{Q}^{\mathcal{N}} = \quad & \sum_{n\in\mathcal{N}_i^C} \sum_{i\in\mathcal{I}} \sum_{r\in\mathcal{R}_i^A} m_n C_{irn}^{A} x_{irn}^{A} && \text{(6.24a)}\\
& + \sum_{n\in\mathcal{N}_i^D} \sum_{i\in\mathcal{I}} \sum_{r\in\mathcal{R}_i^A} m_n C_{i,t(n),r}^{R1} x_{irn}^{R1} && \text{(6.24b)}\\
& + \sum_{n\in\mathcal{N}_i^D} \sum_{i\in\mathcal{I}} \sum_{r\in\mathcal{R}_i^A} m_n C_{i,t(n),r}^{R2} x_{irn}^{R2} && \text{(6.24c)}\\
& + \sum_{n\in\mathcal{N}^H} \sum_{r\in\mathcal{R}} \sum_{v\in\mathcal{V}} m_n C_{t(n),rv}^{O} z_{rnv} && \text{(6.24d)}\\
& + \sum_{n\in\mathcal{N}^H} m_n \left(C_{t(n)}^{PR} e_n^{PR} + C_{t(n)}^{IC} e_n^{IC} + C_{t(n)}^{PC} e_n^{PC}\right) && \text{(6.24e)}
\end{aligned}$$

subject to:

$$x_{i,t(n),r} + x^{R2}_{irn} - \sum_{r' \in \mathcal{R}^A_i} x^{R1}_{ir'n} - \sum_{r' \in \mathcal{R}^A_i \setminus \{r\}} x^{R2}_{ir'n} \leq x^A_{irn}, \qquad i \in \mathcal{I},\ r \in \mathcal{R}^A_i,\ n \in \mathcal{N}_{B_i} \tag{6.25}$$

$$x_{i,t(n),r} + x^{R1}_{ir,p(n)} + x^{R2}_{irn} - \sum_{r' \in \mathcal{R}^A_i} x^{R1}_{ir'n} - \sum_{r' \in \mathcal{R}^A_i \setminus \{r\}} x^{R2}_{ir'n} \leq x^A_{irn}, \qquad i \in \mathcal{I},\ r \in \mathcal{R}^A_i,\ n \in \mathcal{N}^F_i \tag{6.26}$$

$$x_{i,t(n),r} + x^{R1}_{ir,p(n)} \leq x^A_{irn}, \qquad i \in \mathcal{I},\ r \in \mathcal{R}^A_i,\ n \in \mathcal{N}_{(L+1)} \tag{6.27}$$

$$\sum_{i \in \mathcal{I}} P^A_{in} x^A_{irn} + \sum_{j \in \mathcal{J}_n} P^B_{jn} y_{jrn} \leq T^R_{rn} + \sum_{v \in \mathcal{V}} z_{rnv}, \quad r \in \mathcal{R},\ n \in \mathcal{N}^H \tag{6.28}$$

$$\sum_{t \in \mathcal{H}^C_i} \sum_{r \in \mathcal{R}^A_i} x_{itr} = 1, \qquad i \in \mathcal{I} \tag{6.29}$$

$$\sum_{r \in \mathcal{R}^B_{jn}} y_{jrn} = 1, \qquad n \in \mathcal{N}^H,\ j \in \mathcal{J}_n \tag{6.30}$$

$$\sum_{r \in \mathcal{R}^A_i} \left(x_{i,t(n),r} - x^{R1}_{irn} - x^{R2}_{irn} \right) \geq 0, \qquad i \in \mathcal{I},\ n \in \mathcal{N}^D_i \tag{6.31}$$

$$\sum_{r \in \mathcal{R}^A_i} \Big(\sum_{i \in \mathcal{I}^B_n} x_{i,t(n),r} + \sum_{i \in \mathcal{I} \setminus \mathcal{I}^B_n} \big(x_{i,t(n),r} + x^{R1}_{ir,p(n)} \big) \Big) + |\mathcal{J}_n| \leq K^{PR}_{t(n)} + e^{PR}_n, \qquad n \in \mathcal{N}^H \tag{6.32}$$

$$\sum_{i \in \mathcal{I}^{IC}} \sum_{r \in \mathcal{R}^A_i} \sum_{\substack{n' \in \mathcal{N}^P_n: \\ D^{IC}_{in'} + t(n') \geq t(n)}} x^A_{irn'} + |\mathcal{J}^{IC}_n| \leq K^{IC}_{t(n)} + e^{IC}_n, \qquad n \in \mathcal{N}^H \tag{6.33}$$

$$\sum_{i \in \mathcal{I}^{PC}} \sum_{r \in \mathcal{R}^A_i} \sum_{\substack{n' \in \mathcal{N}^P_n: \\ D^{PC}_{in'} + t(n') \geq t(n)}} x^A_{irn'} + |\mathcal{J}^{PC}_n| \leq K^{PC}_{t(n)} + e^{PC}_n, \qquad n \in \mathcal{N}^H \tag{6.34}$$

$$x_{itr} \in \{0,1\}, \qquad i \in \mathcal{I},\ t \in \mathcal{H}_i^C,\ r \in \mathcal{R}_i^A \tag{6.35}$$

$$y_{jrn} \in \{0,1\}, \qquad n \in \mathcal{N}^H,\ j \in \mathcal{J}_n,\ r \in \mathcal{R}_{jn}^B \tag{6.36}$$

$$x_{irn}^{R1} \in \{0,1\}, \qquad i \in \mathcal{I},\ r \in \mathcal{R}_i^A,\ n \in \mathcal{N}_i^D \tag{6.37}$$

$$x_{irn}^{R2} \in \{0,1\}, \qquad i \in \mathcal{I},\ r \in \mathcal{R}_i^A,\ n \in \mathcal{N}_i^D \tag{6.38}$$

$$x_{irn}^{A} \in \{0,1\}, \qquad i \in \mathcal{I},\ r \in \mathcal{R}_i^A,\ n \in \mathcal{N}_i^C \tag{6.39}$$

$$0 \leq z_{rnv} \leq T_{rnv}, \qquad r \in \mathcal{R},\ n \in \mathcal{N}^H,\ v \in \mathcal{V} \tag{6.40}$$

$$e_n^{PR} \geq 0, \qquad n \in \mathcal{N}^H \tag{6.41}$$

$$e_n^{IC} \geq 0, \qquad n \in \mathcal{N}^H \tag{6.42}$$

$$e_n^{PC} \geq 0, \qquad n \in \mathcal{N}^H \tag{6.43}$$

6.4.3 Explanation of the model

The explanation given in Section 6.2.3 for the scenario formulation holds for most parts of the node formulation, as the constraints in the two formulations in general correspond to each other completely. There is, however, a few differences, described below:

The scenario and time indices are for many of the variables and parameters exchanged with a node index. The sets are also adjusted. The time period for a specific node is given as *t(n)*, and the preceding node is given as *p(n)*. The variables x_{itr} are not given with a node index, as they represent the first decision, taken before the planning period starts. They will not be affected by how many emergency patients that arrive, as they are decided before this information is known.

(6.24a) - (6.24e) The total expected costs, calculated in the objective function, are calculated by weighting the cost of decisions at each node by the probability of visiting that node.

(6.33) and (6.34) The third summation mark, summing over a set of nodes, is written to include all elective patients that are in post-op in time period $t(n)$. This depends on the duration of stay at post-op, $D_{in'}^{IC}$ and $D_{in'}^{PC}$.

The non-anticipativity constraints of the scenario formulation are removed in the node formulation, as they become unnecessary.

6.5 Alterations of the model

By changing which recourse decisions are allowed, 4 alterations of the model are defined:

a) Both recourse decisions of type 1 and type 2 are allowed

b) Only recourse decisions of type 1 are allowed (postponing the elective patients one time period)

c) Only recourse decisions of type 2 are allowed (changing operating rooms for the elective patients within the same time period)

d) Neither recourse decision of type 1 nor type 2 are allowed

The basic recourse decision of allocating operating rooms to the arriving emergency patients is always allowed.

6.6 A small example case

In this section, a small example problem and its solution in Xpress^{MP} will be presented to illustrate how the stochastic model works. It is a very simplified version of a complete problem, and the parameters are chosen to keep the problem easy to grasp.

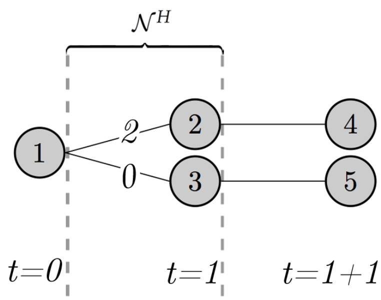


Figure 6.2: Scenario tree illustrating the small example problem

Time periods within the horizon:	1
# of elective patients:	3
# of emergency patients (each with probability 0.5):	0 or 2
# of rooms:	2
Release period, all patients:	1
Duration of surgery, all patients:	4
Regular capacity per OR:	8
Overtime capacity:	0
Capacity pre-op, ICU and PACU:	unlimited
Set of ORs that the emergency patients can go to:	{1}
Set of ORs that elective patient 1 can go to:	{1}
Set of ORs that elective patient 2 and 3 can go to:	{1,2}

Table 6.2: Data values for the small example problem (1/3)

Figure 6.2 shows the scenario tree for the example problem. There is only one time period within the horizon. The objective of the problem is to make a schedule for 3 elective patients at the lowest possible total costs. There is an uncertainty in the problem: either 2 or 0 emergency patients will arrive in time period 1, each with a probability of 0.5 of happening. This leads to two possible scenarios. All elective patients are released in time period 1. The duration of surgery is set to 4 hours for all patients. For the sake of simplicity, no overtime is allowed, and the capacities of pre-op and post-op are set to be unlimited. 2 ORs are available, and the total regular capacity is set to be 8 hours per operating room. The emergency patients and elective patient number 1 can only go to OR 1. Elective patients number 2 and 3 can go to both OR 1 and 2. These values are summarized in Table 6.2.

The costs of performing the elective cases are for time period 1 set as shown in Table 6.3, and for time period 2 (postponing out of horizon) 20 for all patients. The cost of performing recourse decisions of type 1 is for both of the ORs set to be 20 for elective patients 1 and 2, and 5 for elective patient 3. The cost of performing recourse decisions of type 2 is for both of the ORs set to 20 for elective patients 1 and 3, and 5 for elective patient 2. These values are summarized in Table 6.4.

Elective patient #	1	2	3
C^A_{irn} for OR 1	10	10	10
C^A_{irn} for OR 2	10	16	16

Table 6.3: Data values for the small example problem (2/3), C^A_{irn} for $t(n) = 1$

Cost of postponing an elective patient out of horizon, C^A_{irn}, $t(n) = 2$	
for all elective patients:	20
Cost of recourse decisions of type 1, C^{R1}_{itr}	
for elective patients 1 and 2:	20
for elective patient 3:	5
Cost of recourse decisions of type 2, C^{R2}_{itr}	
for elective patients 1 and 3:	20
for elective patient 2:	5

Table 6.4: Data values for the small example problem (3/3)

The node formulation of the model implemented in Xpress$^{\text{MP}}$ solves the problem in 0.03 seconds. The output is formatted as shown in Figure 6.3. The output shows that the expected total costs are minimized to 43. Taking a closer look at the different cost parts of the objective function, what happens in each scenario can be seen. In scenario 1, the number of emergency patients arriving is 2. In this case, three recourse actions will be made:

- Both emergency cases will need to be performed in time period 1 and OR 1. This occupies all the regular capacity of the OR, and as overtime is not allowed, the originally scheduled elective cases need to be moved.
- Elective case 2 will be moved to OR 2, generating a cost of 5 for recourse of type 2
- Elective case 3 will be postponed to the next planning horizon, generating a cost of 5 for recourse of type 1 *and* a postponement cost of 20.

In scenario 2, no emergency patients will arrive. There will be no need to rescheduling the elective patient surgeries that are planned. The total costs of both types of recourse decisions will be 0, and no elective cases will be postponed to the next planning horizon. As each scenario may happen with a probability of 0.5, the costs will be as shown in the figure.

```
Datafile:        examplecase/examplecase.txt
Comments:        V60, max 3600 sec

Output is written to: examplecase/examplecase_Output.txt
The corresponding variable values of the best solution are written to: examplecase/examplecase_FIX.txt

SOLUTION PROCESS:
Time:    Best Bound:    Solution value:
0.016    38             50.5
0.031    42.25          43


Total running time:        0.031

The average expected costs are minimized to: 43
Best bound: 43

Expected costs of..
    ..performing the elective cases: 28
    ..postponing elective cases to the next planning horizon: 10
    ..postponing the elective cases one time period (recourse 1): 2.5
    ..changing the operating rooms of the elective cases (recourse 2): 2.5
    ..total overtime: 0
    ..exceeding the pre-op capacity: 0
    ..exceeding the ICU capacity: 0
    ..exceeding the PACU capacity: 0


TIME PERIOD:   1

    Elective cases scheduled to be done:
    ------------------------------------

    room #     |
              1|    2, 3,
              2|    1,

    Expected overtime hours per price range:
                    1
         room 1:    0
         room 2:    0

    Expected amount of passengers exceeding the capacity of
         pre-op:    0
            ICU:    0
          PACU :    0


Elective cases scheduled to be done in the next planning horizon:
                    0
```

Figure 6.3: Model output when solving the example problem

7. Heuristics applied to the model

In this chapter, the heuristics chosen to be applied to the model described are presented. Relevant theory can be found in Section 4.3. The decisions made regarding the fix and relax heuristic applied are described in Section 7.1, and regarding the improvement heuristic in Section 7.2.

7.1 Fix and relax

A fix and relax heuristic, as described in Section 4.3.1, is applied to the stochastic model given in Chapter 6. The number of time intervals n is set to equal the number of time periods in the horizon of the problem and one period after,
$n = |\mathcal{H}| + 1$.

As mentioned in Section 4.3.1, it can be advantageous to add a stopping criteria to the iterations, as it might not be worth solving each subproblem to optimality (or to the chosen max MIP gap on 0.5%). Ideally, the fix and relax heuristic should be able to find a good (enough) solution faster than during normal runs. As the time limit set for the normal runs is 3 600 seconds, the time limit chosen for the fix and relax heuristic is smaller. The algorithm is chosen to run for 1 200 seconds. Early iterations are given greater time limits than the later, due to the change in subproblem complexity during the algorithm [1]. The time limits set for the test instances are shown in Table 7.1.

$\lvert\mathcal{H}\rvert =$	2	3	4	5
Iteration 1	500	450	420	350
Iteration 2	400	375	360	300
Iteration 3	300	225	240	250
Iteration 4		150	120	150
Iteration 5			60	100
Iteration 6				50
Total	1 200	1 200	1 200	1 200

Table 7.1: Time limits (seconds) set per iteration in the fix and relax heuristic for data instances of different size

The variables chosen to be "fixed and relaxed" are the binary elective variables, x_{itr}. The other binary variables are similarly relaxed for the later time intervals, else kept binary. Table 7.2 gives an overview of the groupings of the binary variables in the different intervals, valid for all the variables that exist for the given time period t. The continuous variables are kept continuous throughout the whole solution process.

Iteration	Fixed binary, {0,1}		Binary, {0,1}		Continuous, [0,1]	
1			x_{itr},	$t = 1$	x_{itr},	$t = 2, \ldots, n$
			x^{A}_{irn},	$t(n) = 1$	x^{A}_{irn},	$t(n) = 2, \ldots, n$
			x^{R1}_{irn},	$t(n) = 1$	x^{R1}_{irn},	$t(n) = 2, \ldots, n$
			x^{R2}_{irn},	$t(n) = 1$	x^{R2}_{irn},	$t(n) = 2, \ldots, n$
			y^{A}_{jrn},	$t(n) = 1$	y^{A}_{jrn},	$t(n) = 2, \ldots, n$
2	x_{itr},	$t = 1$	x_{itr},	$t = 2$	x_{itr},	$t = 3, \ldots, n$
			x^{A}_{irn},	$t(n) = 1, 2$	x^{A}_{irn},	$t(n) = 3, \ldots, n$
			x^{R1}_{irn},	$t(n) = 1, 2$	x^{R1}_{irn},	$t(n) = 3, \ldots, n$
			x^{R2}_{irn},	$t(n) = 1, 2$	x^{R2}_{irn},	$t(n) = 3, \ldots, n$
			y^{A}_{jrn},	$t(n) = 1, 2$	y^{A}_{jrn},	$t(n) = 3, \ldots, n$
$\vdots$	$\vdots$		$\vdots$		$\vdots$	
n	x_{itr},	$t = 1, \ldots, (n-1)$	x_{itr},	$t = n$		
			x^{A}_{irn},	$t(n) = 1, \ldots, n$		
			x^{R1}_{irn},	$t(n) = 1, \ldots, n$		
			x^{R2}_{irn},	$t(n) = 1, \ldots, n$		
			y^{A}_{jrn},	$t(n) = 1, \ldots, n$		

Table 7.2: Variable groups per iteration, fix and relax heuristic

7.2 Improvement algorithm

An improvement algorithm, described in Section 4.3.2, is applied to the best solutions found with different methods. The variables chosen to be fixed are also here the binary elective variables, x_{itr}. They are split into groups of elective cases, depending on the index i, as shown in Table 7.3. For the problems with horizon $|\mathcal{H}| \leq 4$, they are split into 3 groups, and for the problems with horizon $|\mathcal{H}| = 5$ also into 5 groups, to see if it has an effect on the solution or solution time.

# of elective patients	Group 1 of x_{itr}	Group 2 of x_{itr}	Group 3 of x_{itr}
60	$i \in \{1,\ldots,20\}$	$i \in \{21,\ldots,40\}$	$i \in \{41,\ldots,60\}$
90	$i \in \{1,\ldots,30\}$	$i \in \{31,\ldots,60\}$	$i \in \{61,\ldots,90\}$
120	$i \in \{1,\ldots,40\}$	$i \in \{41,\ldots,80\}$	$i \in \{81,\ldots,120\}$
150	$i \in \{1,\ldots,50\}$	$i \in \{51,\ldots,100\}$	$i \in \{101,\ldots,150\}$

Table 7.3: Variable groups to be fixed/unfixed in the improvement heuristic of 3 iterations

The computing time with the improvement heuristic is limited to a maximum of 1 200 seconds per loop through all of the groups defined. This is a relatively low limit, and it is chosen to make the total time for both constructing a solution with the fix and relax heuristic and trying to improve it (one or two loops through all of the elective patients) less than the time limit of 3 600 seconds for normal runs. The time limits per interval are shown in Table 7.4.

	3 iterations	5 iterations
Iteration 1	400	240
Iteration 2	400	240
Iteration 3	400	240
Iteration 4		240
Iteration 5		240
Total	1 200	1 200

Table 7.4: Time limits (seconds) set per iteration for every loop through the improvement heuristic

In this thesis, the improvement algorithm is applied to:

- The best solutions found during normal runs, for the solutions that are not optimal
- The solutions found with the fix and relax heuristic
- The best solutions found during normal runs for alterations *a* and *b*. The solution values found during normal run for alteration *d* are used as initially fixed values.

8. Test instances used in the computational study

In this chapter, the different data sets that have been created for performing the computational tests are presented. The values are chosen to create cases of a realistic size and structure.

In Section 8.1, a list over the needed input to generate a fictive case for the stochastic multi stage problem is provided. The rest of the parameters required by the model are randomly created according to the given probability distributions, with the help of a spreadsheet and the function "$= rand()$", drawing a number between 0 and 1 from the uniform probability distribution. The input is formatted to suit the node formulation, as it is computationally easier to solve than the scenario formulation. In Section 8.2, the parameter values and probability distributions chosen for the different data sets are presented. Last, an overview of the sets of test instances will be given in Section 8.3.

8.1 Input data needed for generating a test instance

The input values needed to generate a test instance for the node formulation in a spreadsheet are as follows:

1. Information about the scenario tree:
   (a) Amount of nodes $|\mathcal{N}|$
   (b) Amount of nodes within the horizon, $|\mathcal{N}^H|$ (all nodes except from the origin node and the leaf nodes)
   (c) The probabilities of visiting the nodes, m_n
   (d) Time period of node n, $t(n)$
   (e) Preceding node of node n, $p(n)$
2. The length of the time horizon, measured in amount of time periods, $|\mathcal{H}|$
3. The number of elective patients to be scheduled, $|\mathcal{I}|$

4. The probability that an elective case is going to ICU rather than PACU for post-op, to create the $\mathcal{I}^{IC}$ and $\mathcal{I}^{PC}$ sets
5. A total count of the non-elective patients arriving during the horizon, for all the possible scenarios (letting a non-elective patient have a separate index for every scenario that he is involved in) $\sum_{n \in \mathcal{N}^H} |\mathcal{J}_n|$
6. Amount of non-elective arrivals in each node (time period/scenario), making the $\mathcal{J}_n$ sets
7. The probability that a non-elective case is going to ICU rather than PACU for post-op, to create the $\mathcal{J}_n^{IC}$ and $\mathcal{J}_n^{PC}$ sets
8. The number of available rooms, $|\mathcal{R}|$
9. The sets of rooms that the different elective patients can go to, $\mathcal{R}_i^A$
10. The sets of rooms that the different emergency patients can go to, $\mathcal{R}_{jn}^B$
11. Distribution of release periods for the elective patients, to create the B_i parameters
12. A probable cost distribution for an elective case, per room number and node, to create the C_{irn}^A parameters
13. A probable cost distribution for recourse decisions of type 1 (postponing an elective patient one time period), per room number and time period, to create the C_{itr}^{R1} parameters
14. A probable cost distribution for recourse decisions of type 2 (changing OR for an elective patient), per room number and time period, to create the C_{itr}^{R2} parameters
15. The costs of overtime per room, time period, and overtime price range, C_{trv}^O
16. The costs of exceeding capacity in pre-op per time period, C_t^{PR}
17. The costs of exceeding capacity in ICU per time period, C_t^{IC}
18. The costs of exceeding capacity in PACU per time period, C_t^{PC}
19. The probabilities of duration of surgery for elective patients, to create the P_{in}^A parameters
20. The probabilities of duration of surgery for non-elective patients, to create the P_{jn}^B parameters
21. The number of overtime price ranges, $|\mathcal{V}|$
22. The regular capacities of the ORs per room and node, T_{rn}^R
23. The overtime capacities of the ORs per room, node, and overtime price range, T_{rnv}
24. The capacities of pre-op per time period, K_t^{PR}

25. The capacities of ICU per time period, K_t^{IC}
26. The capacities of PACU per time period, K_t^{PC}
27. The probabilities of different lengths of stay at post-op for elective patients, to create the D_{in}^{IC} and D_{in}^{PC} parameters
28. The probabilities of different lengths of stay at post-op for non-elective patients, to create the sets $\mathcal{J}_n^{IC}$ and $\mathcal{J}_n^{PC}$

8.2 Chosen parameter values for the test instances

Test instances were created for different structures of the scenario trees. The smallest cases had a planning horizon of 2 time periods, and the largest cases 5 time periods. The time periods are set to equal days. $t = 0$ is meant to represent Sunday evening, $t = 1$ Monday, and so on. The scenario trees for the different test instance series are shown in Figures 8.1 to 8.5. The trees represent the possible scenarios of emergency patient arrivals. The branches between the nodes in $\mathcal{N}^H$ represent the possible realizations of the uncertain parameters; the emergency patient arrivals per day.

The input values *1a-c*, *2*, *3*, and *5*, as described in Section 8.1, for the different series are given in Tables 8.1 to 8.5. The values *1d* (time period of node n) and *1e* (preceding node of node n) are left to be seen in the figures.

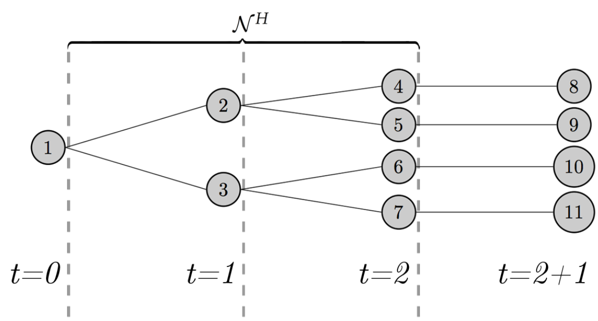


Figure 8.1: Scenario tree illustrating the test instances in series *h2*, input 1

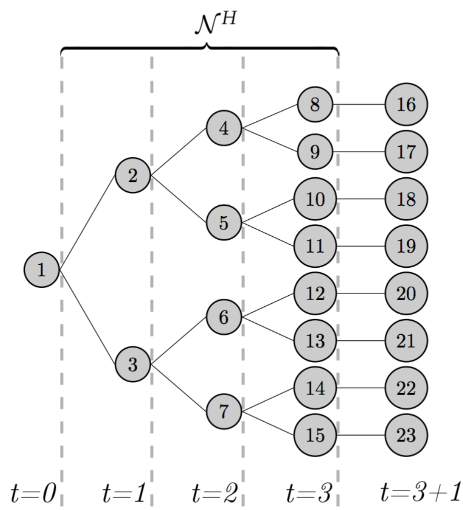


Figure 8.2: Scenario tree illustrating the test instances in series *h3*, input 1

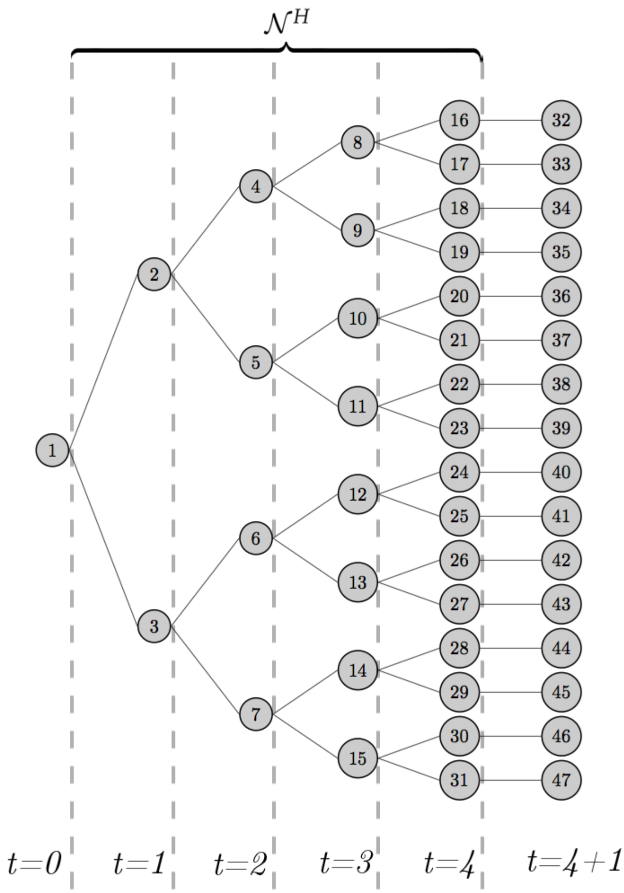


Figure 8.3: Scenario tree illustrating the test instances in series *h4*, input 1

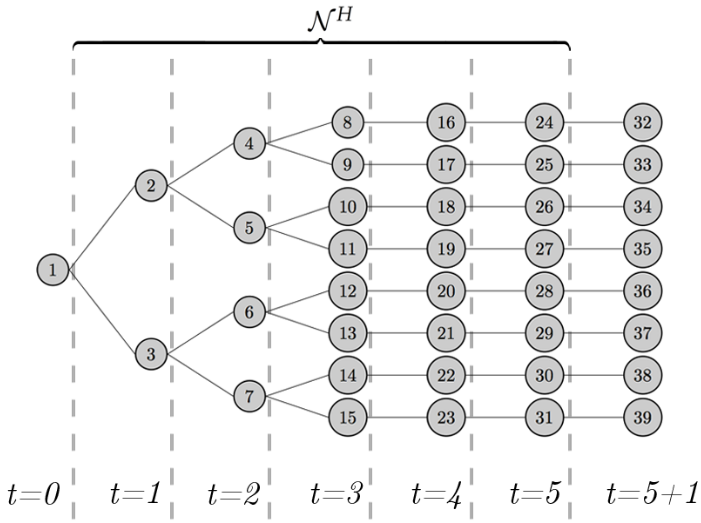


Figure 8.4: Scenario tree illustrating the test instances in series *h5b*, input 1

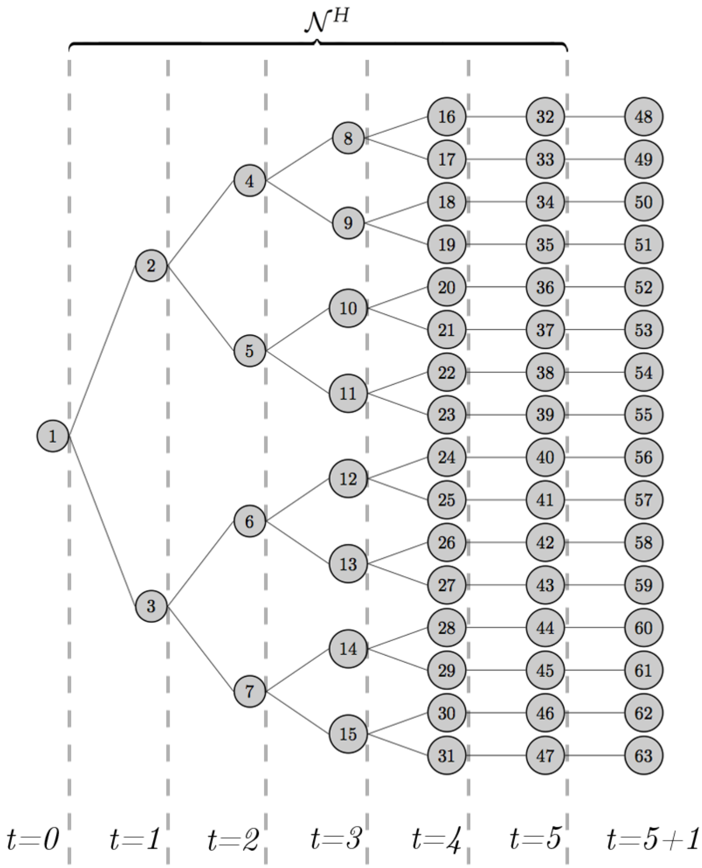


Figure 8.5: Scenario tree illustrating the test instances in series *h5c*, input 1

1a.	$\lvert\mathcal{N}\rvert$	Amount of nodes	11
1b.	$\lvert\mathcal{N}^H\rvert$	Amount of nodes in the horizon	6 (nodes 2 - 7)
1c.	$m_n, n \in [2,3]$	Probability of visiting nodes 2 - 3	0.5
	$m_n, n \in [4,11]$	Probability of visiting nodes 4 - 11	0.25
2.	$\lvert\mathcal{H}\rvert$	Horizon	2 days
3.	$\lvert\mathcal{I}\rvert$	# of elective cases	60
5.	$\sum_{n\in\mathcal{N}^H} \lvert\mathcal{J}_n\rvert$	Size of non-elective case counter	18
	$\lvert\mathcal{S}\rvert$	# of scenarios	4

Table 8.1: Test instance series *h2*, input 1, 2, 3, 5

1a.	$\lvert\mathcal{N}\rvert$	Amount of nodes	23
1b.	$\lvert\mathcal{N}^H\rvert$	Amount of nodes in the horizon	14 (nodes 2 - 15)
1c.	$m_n, n \in [2,3]$	Probability of visiting nodes 2 - 3	0.5
	$m_n, n \in [4,7]$	Probability of visiting nodes 4 - 7	0.25
	$m_n, n \in [8,23]$	Probability of visiting nodes 8 - 23	0.125
2.	$\lvert\mathcal{H}\rvert$	Horizon	3 days
3.	$\lvert\mathcal{I}\rvert$	# of elective cases	90
5.	$\sum_{n\in\mathcal{N}^H} \lvert\mathcal{J}_n\rvert$	Size of non-elective case counter	42
	$\lvert\mathcal{S}\rvert$	# of scenarios	8

Table 8.2: Test instance series *h3*, input 1, 2, 3, 5

1a.	$\lvert\mathcal{N}\rvert$	Amount of nodes	47
1b.	$\lvert\mathcal{N}^H\rvert$	Amount of nodes in the horizon	30 (nodes 2 - 31)
1c.	$m_n, n \in [2,3]$	Probability of visiting nodes 2 - 3	0.5
	$m_n, n \in [4,7]$	Probability of visiting nodes 4 - 7	0.25
	$m_n, n \in [8,15]$	Probability of visiting nodes 8 - 15	0.125
	$m_n, n \in [16,47]$	Probability of visiting nodes 16 - 47	0.0625
2.	$\lvert\mathcal{H}\rvert$	Horizon	4 days
3.	$\lvert\mathcal{I}\rvert$	# of elective cases	120
5.	$\sum_{n\in\mathcal{N}^H} \lvert\mathcal{J}_n\rvert$	Size of non-elective case counter	90
	$\lvert\mathcal{S}\rvert$	# of scenarios	16

Table 8.3: Test instance series *h4*, input 1, 2, 3, 5

1a.	$\lvert\mathcal{N}\rvert$	Amount of nodes	39
1b.	$\lvert\mathcal{N}^H\rvert$	Amount of nodes in the horizon	30 (nodes 2 - 31)
1c.	$m_n, n \in [2,3]$	Probability of visiting nodes 2 - 3	0.5
	$m_n, n \in [4,5]$	Probability of visiting nodes 4 - 7	0.25
	$m_n, n \in [8,39]$	Probability of visiting nodes 8 - 39	0.125
2.	$\lvert\mathcal{H}\rvert$	Horizon	5 days
3.	$\lvert\mathcal{I}\rvert$	# of elective cases	150
5.	$\sum_{n\in\mathcal{N}^H} \lvert\mathcal{J}_n\rvert$	Size of non-elective case counter	90
	$\lvert\mathcal{S}\rvert$	# of scenarios	8

Table 8.4: Test instance series *h5b*, input 1, 2, 3, 5

1a.	$\lvert\mathcal{N}\rvert$	Amount of nodes	63
1b.	$\lvert\mathcal{N}^H\rvert$	Amount of nodes in the horizon	46 (nodes 2 - 47)
1c.	$m_n, n \in [2,3]$	Probability of visiting nodes 2 - 3	0.5
	$m_n, n \in [4,7]$	Probability of visiting nodes 4 - 7	0.25
	$m_n, n \in [8,15]$	Probability of visiting nodes 8 - 15	0.125
	$m_n, n \in [16,63]$	Probability of visiting nodes 16 - 63	0.0625
2.	$\lvert\mathcal{H}\rvert$	Horizon	5 days
3.	$\lvert\mathcal{I}\rvert$	# of elective cases	150
5.	$\sum_{n\in\mathcal{N}^H} \lvert\mathcal{J}_n\rvert$	Size of non-elective case counter	138
	$\lvert\mathcal{S}\rvert$	# of scenarios	16

Table 8.5: Test instance series *h5c*, input 1, 2, 3, 5

Number of patients to be treated in the horizon

The number of *elective cases* is chosen to make an average of 30 cases per day within the horizon. This number is based on a case with 4 available operating rooms, as suggested by Atle Riise in Sintef [22].

The number of *non-elective cases* to be treated within the horizon will vary between the scenarios. An emergency patient that will arrive in two scenarios will be included two sets of emergency cases, $\mathcal{J}_n$, and count twice in the size of non-elective case counter, $\sum_{n\in\mathcal{N}^H} \lvert\mathcal{J}_n\rvert$. This size of this counter must therefore not be confused with a total of emergency patients arriving. In the test instances, its value is based on an average of about 3 emergency patient arrivals per day per possible scenario ($3 \cdot$ [# of branches in the scenario tree]). The arrival numbers per node are drawn randomly.

Post-op distribution

The probability that an *elective case* is going to ICU rather than PACU for post-op is set to be 20%. This is on average every fifth patient. The reason for the choice of percentage is that it is more likely that a patient is going through an uncomplicated procedure than a procedure that needs extra resources at post-op. The probability that a *non-elective case* is going to ICU rather than PACU for post-op is set to be 33%. This is on average every third patient. This percentage is chosen to be higher than for the elective patients, as it is assumed that, on average, the non-elective patients are in a less stable health situation than the elective patients. The probability distributions of post-op destinations are given in Table 8.6.

Probability of going to	ICU	PACU
For elective patients	20%	80%
For non-elective patients	33%	67%

Table 8.6: Distribution of post-op destinations, input 4, 7

Sets of rooms that the patients can be treated in

Four operating rooms are set to be available in the test instances, $|\mathcal{R}| = 4$. This could be realistic for a medium-sized hospital in Norway [22]. In many hospitals, the operating room capacity is split between different hospital sections (such as cardiology, neurology etc) [22]. Every section is allotted given time periods and operating rooms, according to a schedule set. In the test instances generated, the patients are split into four groups, to simulate that they belong to different sections of the hospital. As a simplification, these groups are only set to have limitations on which rooms they may go to, not on which time periods (although the model also allows for setting limitations on time periods, by adjusting the set of nodes that the patients may be scheduled to). The elective patients are divided into the four groups according to the percentages given in Table 8.7.

% of elective patients	Group #	Set of ORs, $\mathcal{R}_i^A$
40%	1	{ 1 2 3 }
20%	2	{ 3 4 }
20%	3	{ 1 2 4 }
10%	4	{ 4 }

Table 8.7: Sets of ORs available for the elective patients, input 8, 9

The emergency patients of the test instances are set to be able to go to all rooms, as shown in Table 8.8. This is based on an assumption that they will be performed surgery on as soon as possible, no matter which section they belong to.

% of emergency patients	Group #	Set of ORs, $\mathcal{R}_i^A$
100%	-	{ 1 2 3 4 }

Table 8.8: Sets of ORs available for the emergency patients, input 10

Release periods

The release period for an elective patient is set according to the probability distributions given in Table 8.9. Patients released in the week-end before the planning horizon, would in the model get their release date on Monday. This is done as it is likely to believe that the surgery capacity in the week-ends mainly is reserved for the non-elective patients. The probability of having a release date on day 1, Monday, is therefore set to be larger than for day 2. It is also done to include potential postponed (queued) elective patients from the last planning period.

Series	Release period	1	2	3	4	5
h2	For elective patients	65%	35%			
h3	For elective patients	45%	35%	20%		
h4	For elective patients	40%	30%	20%	10%	
h5b	For elective patients	30%	30%	15%	15%	10%
h5c	For elective patients	30%	30%	15%	15%	10%

Table 8.9: Distribution of release periods for the elective patients, input 11

Cost of performing elective cases

The cost of performing an elective case is drawn from a discrete uniform distribution set to be in the intervals given in Table 8.10, varying with the horizon.

Time period:	All days within the horizon	Postponed
Rooms:	[1 2 3 4]	[1 2 3 4]
Costs, C_{irn}^A :	100 ± 5	175 ± 25

Table 8.10: Costs of performing the elective cases, input 12

Cost of recourse decisions

The cost of making a recourse decision of type 1 (R1) or type 2 (R2) is for all the elective cases set to be varying around 15 or 5, respectively, for all the horizons and rooms. These values are shown in Table 8.11.

13	C_{itr}^{R1}	Cost of R1	15 ± 2	for all i, t, r
14	C_{itr}^{R2}	Cost of R2	5 ± 2	for all i, t, r

Table 8.11: Costs of recourse decisions, input 13, 14

Overtime

The regular capacities of the ORs are set to be 10 hours for all rooms. This could for example represent a working day from 8 am to 6 pm. The overtime cost is for the test instances split into three price ranges (3, 3 and 8 hours). This could illustrate one price range from 6 pm to 9 pm, another from 9 pm to midnight, and a third one for from midnight to 8 am. They are set to be equal between the rooms, this is done for simplicity. The ranges are given in Table 8.12.

21	$\lvert\mathcal{V}\rvert$	# of overtime price ranges	3	for all i, t, r
22	T_{rn}^{R}	Regular capacities	10	for all r, n
23	T_{rnv}^{R}	Overtime capacities:		
	T_{rn1}^{R}	..of price range 1	3	for all r, n
	T_{rn2}^{R}	..of price range 2	3	for all r, n
	T_{rn3}^{R}	..of price range 3	8	for all r, n

Table 8.12: Overtime price ranges, input 21 - 23

The costs of the different ranges are given in Table 8.13.

	v	Overtime price range:	1	2	3
15	C_{trv}^{O}	Cost overtime (for all rooms):	35	60	100

Table 8.13: Costs of overtime, input 15

Cost of exceeding capacity in pre-op and post-op

For all time periods, the cost of exceeding capacity is set to be 10 for pre-op and PACU, and 20 for ICU, as given in Table 8.14.

			(for all time periods)
16	Cost of exceeding the capacity of Pre-Op	C_t^{PR}	10
17	Cost of exceeding the capacity of ICU	C_t^{IC}	20
18	Cost of exceeding the capacity of PACU	C_t^{PC}	10

Table 8.14: Costs of exceeding capacity in pre-op and post-op, input 16, 17, 18

Duration of surgery

The duration of the surgery is set to be between 0.5 and 4.5 hours as shown in Figure 8.6 for the elective patients and Figure 8.7 for the non-elective patients. It is assumed that the parameter has a larger spread and a larger average for the non-elective patients than for the elective patients.

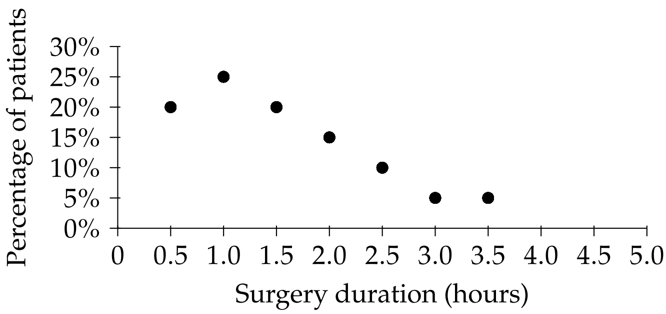


Figure 8.6: Duration of surgery for elective patients, input 19

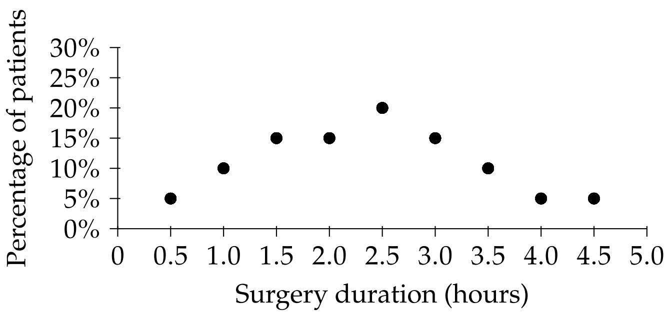


Figure 8.7: Duration of surgery for non-elective patients, input 20

Capacities of pre-op and post-op

The capacities are set to be 35, 10 and 35 for pre-op, ICU and PACU, respectively, see Table 8.15. The reason for setting the ICU capacity low is that it is expensive to reserve resources and keep capacity up for intensive care. The post-op units of real hospitals do sometimes have a lower capacity than the amount of patients [22]. The capacities of the ICU and the PACU are therefore set low enough to create a bottleneck in some scenarios.

			(for all time periods)
24	Capacity of Pre-Op	K_t^{PR}	35
25	Capacity of ICU	K_t^{IC}	10
26	Capacity of PACU	K_t^{PC}	35

Table 8.15: Capacities of pre-op and post-op, input 24, 25, 26

Duration of stay at post-op

When calculating the available capacity of post-op, it is necessary to know for how long the patients are staying at post-op. Some will leave the same day as they arrive, for these patients, the duration of post-op is set to 0. The percentages, shown in Table 8.16, are chosen as the probabilities of staying none, 1, 2, 3, or 4 days. The non-elective patients are assumed to, on average, stay longer at post-op than the elective patients.

Duration post-op (in days)	0	1	2	3	4
For elective patients	40%	30%	20%	10%	0%
For non-elective patients	20%	20%	30%	20%	10%

Table 8.16: Probabilities of length of stay at post-op, input 27, 28

8.3 Overview of the sets of test instances

The sets of test instances used in the computational testing are all generated as described in the previous section. The specifications of the sets are described below. Each set was generated for different horizons, as illustrated in Figures 8.1-8.5.

r4 With 4 ORs available, generated for the different horizons $h2$, $h3$, $h4$, $h5b$, and $h5c$, and the node formulation of the stochastic multi-stage model

r3 With 3 ORs available, generated for the different horizons $h2$, $h3$, $h4$, $h5b$, and $h5c$, and the node formulation of the stochastic multi-stage model

r4-v With 4 ORs available, generated for the horizons $h2$ and $h3$, for both the node formulation of the stochastic multi-stage model and the deterministic model

(the expected average values of the possible scenarios in the stochastic problem)

r4-r With 4 ORs available, generated for the horizon *h*2, and the node formulation of the stochastic multi-stage model (10 test instances).

A collection of test instances of the same horizon size is in this thesis called *test instance series*.

The numbers of variables and constraints for the different test instances generated are shown in Table 8.17. The most complex test instance series are *h*4 and *h*5*c*, consisting of 16 possible scenarios each. The values given in the table are only given for variation *a*, allowing both types of recourse decisions, as described in Section 6.5.

The corresponding values for the other variations are related to the values given for alteration *a* in the following ways: The number of binary variables is reduced with about 25% from alteration *a* to alteration *b* and *c*, and with about 50% to alteration *d*. The number of nonzero elements is reduced with about 20% from alteration *a* to alteration *b*, about 33% to alteration *c* and about 50% to alteration *d*. The number of constraints are the same for all variations.

Instance	Variation	Size horizon	# Scenarios	# Constraints	# variables	# binary variables	# nonzero elements in matrix
r4_h2	a	2	4	2 056	3 551	3 461	14 223
r4_h3	a	3	8	6 510	11 301	11 091	49 675
r4_h4	a	4	16	18 360	32 121	31 581	150 247
r4_h5b	a	5	8	18 356	34 089	33 639	150 845
r4_h5c	a	5	16	30 712	56 338	55 648	272 622
r3_h2	a	2	4	1 990	3 422	3 332	13 654
r3_h3	a	3	8	6 178	10 502	10 292	45 506
r3_h4	a	4	16	17 282	30 032	29 582	138 560
r3_h5b	a	5	8	17 462	31 683	31 233	138 560
r3_h5c	a	5	16	28 800	51 287	50 597	242 747
r4-v_h2	b (det.)	2	4	683	1 186	1 156	3 991
r4-v_h2	b (stoch.)	2	4	2 926	2 122	12 278	2 836
r4-v_h3	b (det.)	3	8	1 238	2 150	2 105	7 760
r4-v_h3	b (stoch.)	3	8	8 695	6 580	40 309	8 485
r4-r_h2	b (average)	2	4	2 096	2 860	2 770	11 845

Table 8.17: Model dimensions of the test instances generated

9. Computational study

In this chapter, the results obtained from solving the test instances described in Chapter 8 will be presented and discussed. Section 9.1 describes how the model was implemented. In Section 9.2 the model is tested at a technical level, varying parameter values and applying heuristics. In Section 9.3 the value of applying the recourse models is studied, and in Section 9.4. Last, in Section 9.5, one potential use of the model is tested.

9.1 Implementation

The implementation was done for the node formulation of the model, given in Section 6.4. This formulation is chosen over the scenario formulation, as it should be computationally easier to solve, with fewer variables and fewer constraints. All relevant files (code, input, output, and data file generators) can be found in the enclosed zip archive or for a limited time at:

`http://folk.ntnu.no/sofiedov/TIO4905`

Hardware

All of the runs were done on nodes in a Linux cluster. The specifications of the nodes are:

Hewlett Packard dl160 G5 PC
2 x Intel QuadCore E5472 3.0 GHz
16 Gb RAM
72 Gb SAS 15000 rpm.

Software

The code was written in the Mosel language and implemented in Xpress-IVE, Version 1.22.04. The solver package used was FICO™ Xpress Optimization Suite by Dash Optimization. The complete codes for the node formulation, the fix and relax heuristic and the improvement heuristic can be found in the documentation folder.

Comments on the Mosel code

After the codes were completed, a tuning was run to reduce the running time, resulting in the following adjustments:

```
setparam("XPRS_DEFAULTALG",3);
setparam("XPRS_CUTFREQ",2);
setparam("XPRS_HEURDIVESTRATEGY",5);
setparam("XPRS_HISTORYCOSTS",0);
setparam("XPRS_MIPPRESOLVE",3);
setparam("XPRS_PERTURB",0.0001);
setparam("XPRS_ROOTPRESOLVE",1);
setparam("XPRS_SBEFFORT",0.25);
setparam("XPRS_SBESTIMATE",5);
```

A max running time stopping criteria was for most of the runs set to 3 600 seconds. In addition, a MIP gap stopping criteria of 0.5% was used in most of the runs, except from in the runs that it was considered of importance to get even more precise results.

Generation of the test instances

Spreadsheets in MS Excel were used to generate the test instances described in the previous chapter. The spreadsheets are written in a way that is suited for Xpress^{MP} and the node formulation code. After completion, the data was saved to a standard *.txt*-file. The spreadsheets and text files can be found in the documentation folder.

Processing of results

The results were written to text files, and are gathered in spreadsheets for the ease of reading and handling. These spreadsheets can also be found in the documentation folder.

9.2 Computational testing

At a technical level, tests were done to find the impact of the recourse decisions on the model, and the potential benefits from applying the fix and relax and the improvement heuristics.

9.2.1 The impact of the recourse decisions

The problem was first run to see how the recourse decisions affect the solutions and solution process. The set of test instances, *r4*, with 4 available ORs were run for the 4 model alterations described in Section 6.5. The set includes test instances of different sizes, with horizons from 2 - 5 days and 4 - 16 possible scenarios.

In alterations *a*, both recourse decisions of type 1 and type 2 were allowed. Only the recourse decisions of type 1, regarding postponement of the elective patients one time period, were allowed in alterations *b*, and only recourse decisions of type 2, changing operating rooms for the elective patients within the same time period, were allowed in alterations *c*. Neither recourse decisions of type 1 nor type 2 were allowed in alteration *d*. The basic recourse decision of allocating operating rooms to the arriving emergency patients were allowed in all alterations.

The results of the runs are shown in Table 9.1. The first column presents the test instance number and the length of the planning horizon. For the complex test instance *r4_h5c_a*, the running time is the time until the first feasible solution was found. The gap in the column to the right in the table is calculated as:
gap=(objective value - best bound)/objective value.

The runs on alteration *d* were, for all the test instances, the fastest to solve. This is because these problems are smaller and less complex than the other alterations.

Alterations *a* and *c* seem to be harder to solve. When reaching the time limit of one hour, the gaps from the best solution found to the best bound were large for these alterations of all the test instances. This indicates that the recourse decisions of type 2 (changing room for an elective patient within the same time period) create difficulties for the solution process. For the most complex test instance, *r4_h5c*, the solver did not find a feasible solution within the time limit for alteration *a*.

The runs on alteration *b* gave better results. Within the time limit of one hour, the least complex test instances, *r4_h2*, *r4_h3*, and *r4_h5b*, were solved to solutions relatively close to the best bound.

Instance	Variation	Running time (sec)	Best bound	Objective value	Gap
r4_h2	a	3 600	3 750	6 715	44.2 %
r4_h2	b	121	6 407	6 440	0.5 %
r4_h2	c	3 600	4 324	6 609	34.6 %
r4_h2	d	< 1	6 518	6 535	0.3 %
r4_h3	a	3 601	4 568	10 493	56.5 %
r4_h3	b	3 600	9 595	9 954	3.6 %
r4_h3	c	3 600	4 274	10 532	59.4 %
r4_h3	d	16	10 085	10 134	0.5 %
r4_h4	a	3 603	4 124	10 248	59.8 %
r4_h4	b	3 602	8 191	10 107	19.0 %
r4_h4	c	3 601	3 391	9 641	64.8 %
r4_h4	d	2	9 475	9 513	0.4 %
r4_h5b	a	3 601	3 561	5 399	34.0 %
r4_h5b	b	3 601	5 118	5 345	4.2 %
r4_h5b	c	3 600	4 843	5 403	10.4 %
r4_h5b	d	< 1	5 443	5 444	0.0 %
r4_h5c	a	15 972	6 028	28 600	78.9 %
r4_h5c	b	3 604	9 798	25 920	62.2 %
r4_h5c	c	3 601	8 149	23 816	65.8 %
r4_h5c	d	2	23 253	23 360	0.5 %

Table 9.1: Results for the test instance set *r4* with 4 ORs available, alterations on allowed recourse decisions

A possible reason for that allowing only recourse decisions of type 1 makes the problem easier to solve than allowing recourse decisions of type 2 could have a connection to the model formulation, given in Section 6.4. The recourse variables of the two types of decisions appear in the same constraints, except from in constraints (6.27) and (6.32), where only the recourse variables of type 1 occur. By allowing these variables, the formulation gets tighter, and there is a possibility that this has an influence on the solution process and solution time.

The runs on alteration *d* gave good results for all the runs. On average, they gave the second best objective values between the alterations for a test instance, in a short time. If the optimal solutions were found for all the alterations, the runs on alteration *d* would be expected to give the worst objective values. Each of the other alterations, *a*-*c*, may be reduced to alteration *d* if all of the recourse variables are set to 0. However, allowing extra recourse decisions makes the problem a lot more complex. With large test instances, the problem gets too complex to solve to a good solution within the time limit of one hour, when allowing recourse decisions of type 1 and 2, especially type 2.

For the smaller test instances, the runs on alteration *b* gave the best objective values between the alterations. For the test instances *r4_h4* and *r4_h5c*, the alteration *d* gave the best objective values.

Figure 9.1 presents how the best solution values approach the best bound values for the different alterations of *r4_h2*, plotted against computational time, given in a logarithmic scale. The figure shows that the gap between best bound and best solution decreases rapidly for the first seconds of all of the alterations. This behavior is common for the runs on all the test instances. The best solution values for alteration *a* and *c* do not at any point in time, within the max time limit, get better than the optimal solution of alteration *d*. Alteration *b* is also solved to optimality, on a value just below the optimal value of alteration *d*.

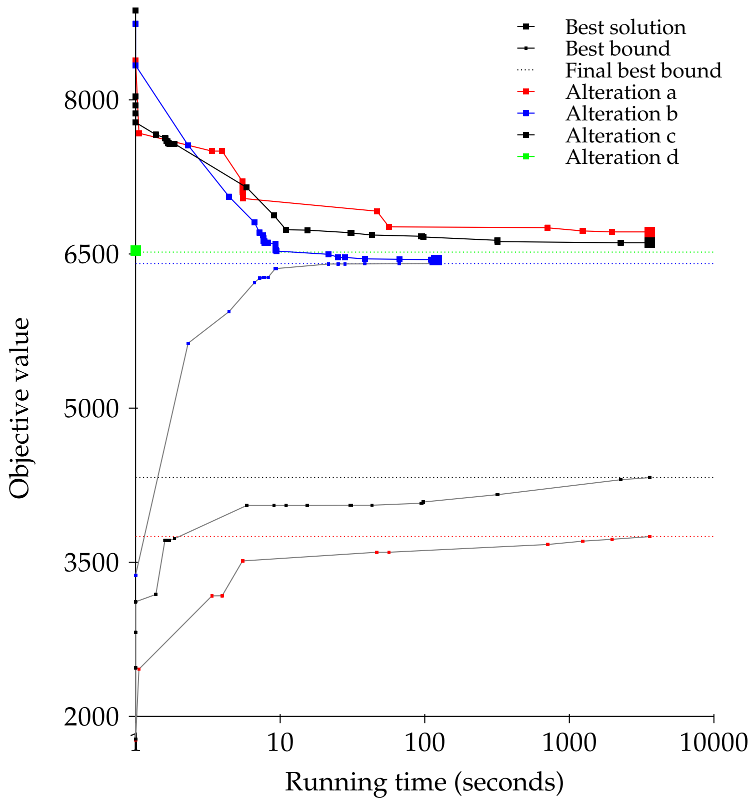


Figure 9.1: Results for test instance *r4_h2*, best bound and solution development

Reducing the amount of ORs to 3 In the test instance set *r4*, the amount of operating rooms available was set to 4. In the test instance set *r3*, the amount is decreased to 3 rooms. The the results of the runs on test instance set *r3* can be found in Tables A.1 and A.2 in Appendix A. The tendencies of the results are the same as for the results on test instance set *r4*. Alteration *b* gives good results for the smaller test instances, but complicates the solution process maybe too much for the complex test instances *r3_h4* and *r3_h5c* to produce a good solution within an hour. Alterations *d* give fairly good results within a short time, on average the second best objective values between the alterations. Alterations *a* and *c* are hard to solve to optimality, and for the two most complex test instances, *r3_h4* and *r3_h5c*, no feasible solutions are found within the time limit for the runs on alteration *a*.

The results do not indicate a clear relation between adjusting the number of available rooms from 4 to 3 and the performance in terms of computational time and gap between best solution and best bound for the different alterations. The gaps in the results of *r3* were on average a bit lower than the gaps in the results of *r4*, but it does not seem to be a consistent trend, for example in one specific alteration. The gaps between best bound and best solution found are compared in Table 9.2.

Series:	*h2*				*h3*				*h4*			
Var.:	a	b	c	d	a	b	c	d	a	b	c	d
Set *r4*	44.2%	0.5%	34.6%	0.3%	56.5%	3.6%	59.4%	0.5%	59.8%	19.0%	64.8%	0.4%
Set *r3*	40.9%	0.4%	44.8%	0.4%	44.4%	0.9%	50.6%	0.5%	58.2%	35.3%	54.2%	0.5%

Series:	*h5b*				*h5c*			
Var.:	a	b	c	d	a	b	c	d
Set *r4*	34.0%	4.2%	10.4%	0.0%	78.9%	62.2%	65.8%	0.5%
Set *r3*	54.4%	2.5%	12.4%	0.1%	68.8%	55.7%	48.0%	0.4%

Table 9.2: Gaps for the test instances in sets *r4* and *r3*.

9.2.2 Applying heuristics

In this section, the value of the two heuristics described in Chapter 7 will be tested and discussed. The fix and relax heuristic is applied to alterations *a* and *b* of every test instance, and the improvement heuristic is applied to the benchmark values that did not reach optimal solution within the time limit set, and to the solutions generated by the fix and relax heuristic. The solution values found during normal runs are denoted *benchmark values*, as they are compared to the solutions found with other methods.

The objective function values found by the different methods compared to the solution time are presented in Figures 9.2-9.6. Note that it is used a logarithmic scale on the horizontal running time axis. The final best bounds of alteration *b* and *d* are shown as the blue and green horizontal lines, respectively. The values of the best bounds of alteration *a* and *c* lie below the areas shown. The filled squares show the benchmark values found, and the non-filled squares show the objective values found after applying the improvement algorithm to the benchmark values. The filled circles show the objective values found with the fix and relax heuristic (with an intended max running time of 1200 seconds), and the non-filled circles show its improved values. The crosses represent the results from the improvement algorithm applied to alterations *a* and *b*, using initial variable values found in the benchmark runs on alteration *d*.

A complete overview of the results can be found in Tables A.1-A.4 in Appendix A. In the tables, the runs are grouped by test instance, variation, whether a benchmark, a fix and relax or an improvement run was done, amount of iterations in the heuristics and max running time. All time values are measured in seconds. The result "nbs" indicates that no better solutions were found using the improvement algorithm. The improvement is calculated as:
improvement=(initial objective value - improved objective value)/initial objective value.

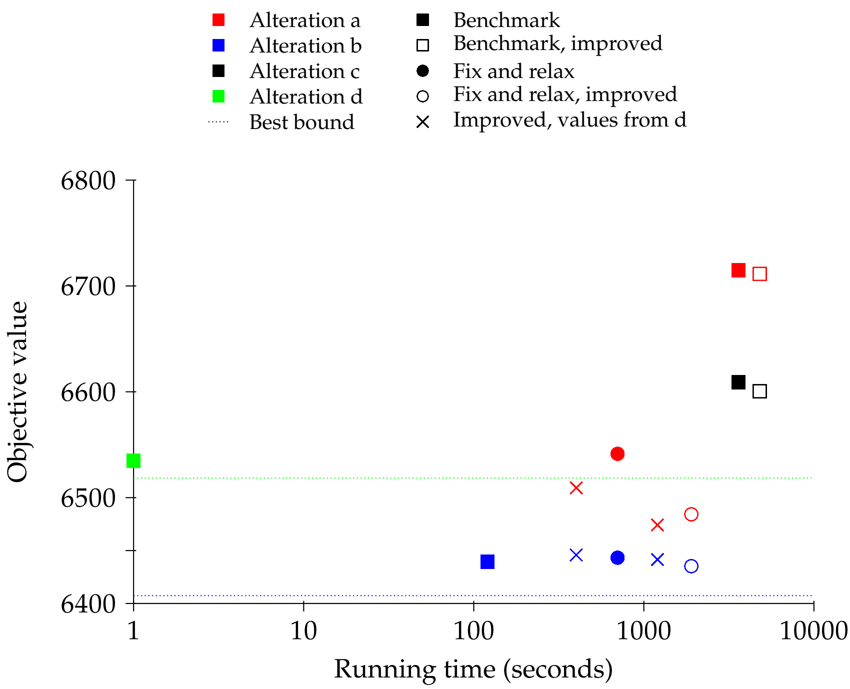


Figure 9.2: Results for test instance *r4_h2*

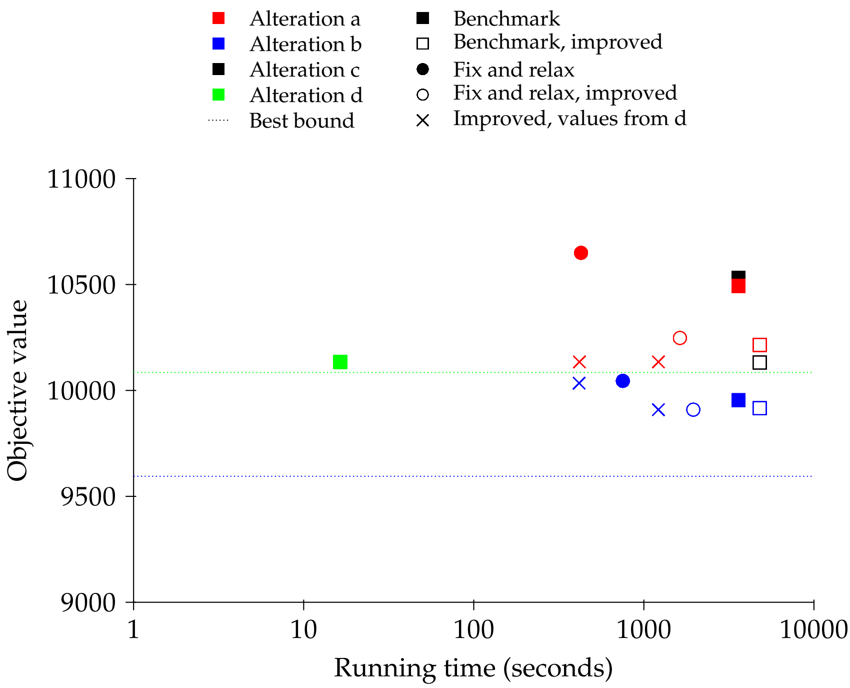


Figure 9.3: Results for test instance *r4_h3*

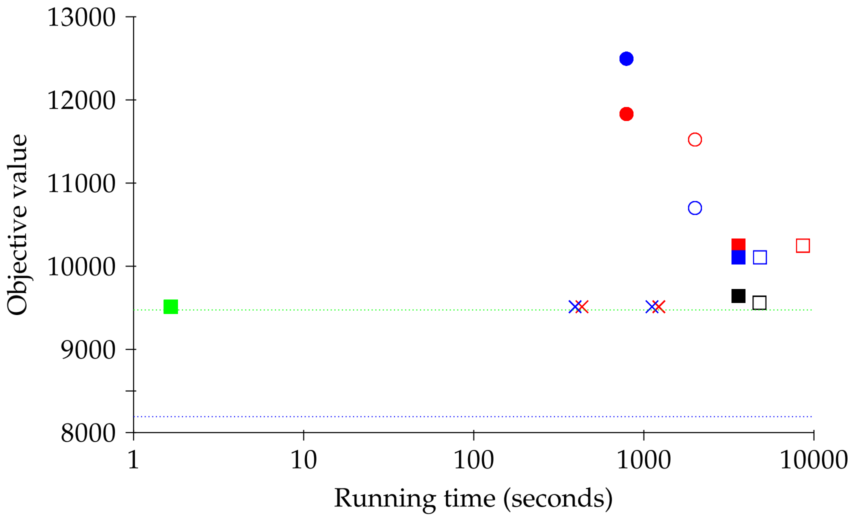


Figure 9.4: Results for test instance *r4_h4*

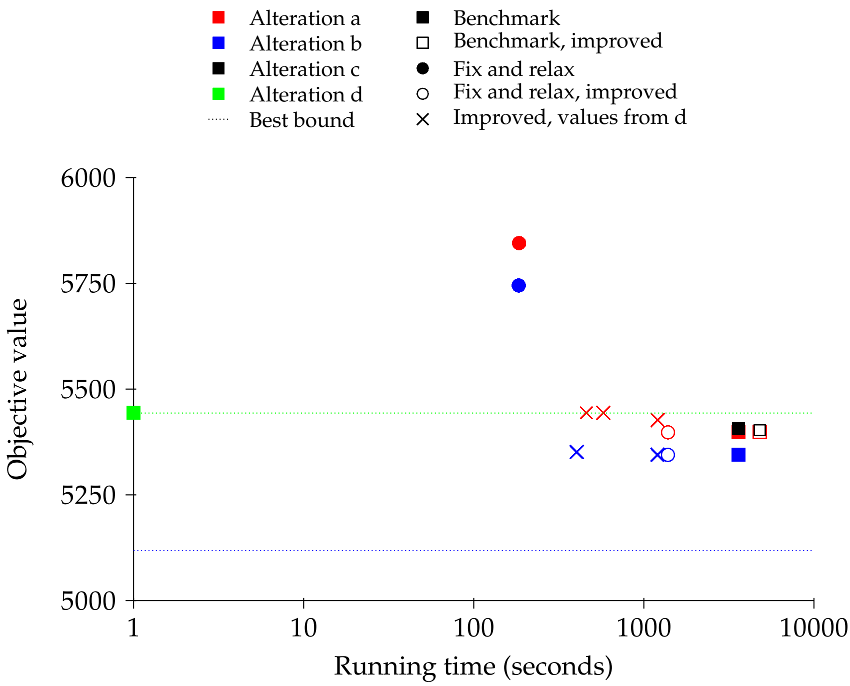


Figure 9.5: Results for test instance *r4_h5b*

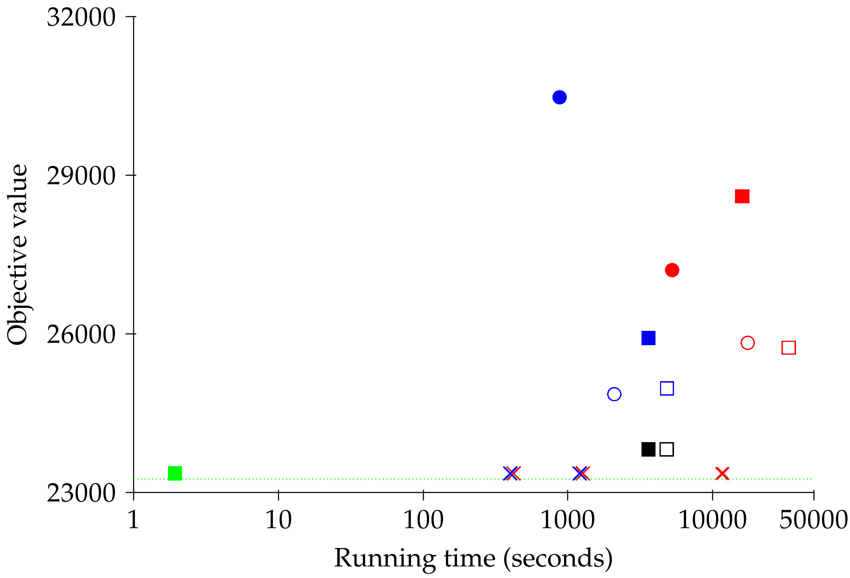


Figure 9.6: Results for test instance *r4_h5c*

Results from the fix and relax heuristic

The runs done with the fix and relax heuristic seem to give moderate good solutions within the time limit of 1200 seconds. For the test instances *r4_h2_a* and *r4_h5c_a*, the heuristic gives better solutions than the benchmark does, and also faster. For most of the other runs, the heuristic gives a poorer solution than the benchmark does, but it seems to produce a good start solution for the improvement algorithm applied, as discussed in the next subsection.

Time limits and MIP gap limits are set as stopping criteria. The time limits per iteration are set as described in Chapter 7. For all of the runs, the time limit was the active stopping criteria. The fix and relax method seems to be fairly consistent in its computational time consumption, as it has to go through all iterations, no matter how easy the problem is to solve, making it an unnecessary heuristic for the problems requiring less than 1200 seconds to find optimal solution.

Results from the improvement heuristic

The improvement heuristic was mainly run with 3 groups of variables to be fixed/unfixed. For the test instances of horizon 5, the heuristic was also run for 5 iterations. These runs did not give any better objective values than the runs with 3 iterations, and the ideal number of iterations for the heuristic is not studied further in this thesis.

Applied to the benchmark solutions The improvement heuristic gives either a small or no improvement when applied to the benchmark solutions of the test instances in the two sets *r4* and *r3*. The largest improvement was measured to 29.3% on test instance *r3_h5b_a*. For the alterations *a*, the improvement heuristic succeeded in improving the benchmark solutions in 40% of the runs. For the improved runs, the heuristic improved the solution with, on average, 10.5% (this value is influenced by the single high improvement measured to 29.3%. Without this improvement, the average value would be 4.2%). For the alterations *b* and *c*, the improvement heuristic gave improvement in 56% and 50% of the runs, improving the solutions, on average, 0.9% and 1.3%, respectively.

Table 9.3 shows both the objective values found by letting the time limit of the normal runs be 4800 seconds and the objective values found with the improvement heuristic applied to the benchmark solutions, with max time $3600 + 1200 = 4800$ seconds. In the test instance set *r4*, the improvement heuristic succeeds in improving the solution oftener than letting the normal model run for a longer time. The complex test instance *r4_h4* does, however, obtain better solutions with a longer normal run than with the heuristic, for all alterations. It can seem like the improvement heuristic is more effective for the smaller test instances than the larger. For test instance *r4_h5c_a*, the improvement with the heuristic was calculated to 10.0%, but at the cost of a very large computational time, far beyond the time limit (allowed if no feasible solutions for an iteration are found within the time limit). A reason for the large improvement on this solution could be that the initial solution was the first and only solution found for the benchmark run, also violating the time limit.

	Objective value			**Improvement percentage***	
Instance	3 600 sec, benchmark	4 800 sec normal run	Improved benchmark	4 800 sec normal run	Improved benchmark
r4_h2_a	6 715	6 699	6 711	0.2 %	0.1 %
r4_h2_c	6 609	6 609	6 601	-	0.1 %
r4_h3_a	10 493	10 493	10 215	-	2.7 %
r4_h3_b	9 954	9 945	9 916	0.1 %	0.4 %
r4_h3_c	10 532	10 532	10 132	-	3.8 %
r4_h4_a	10 248	10 086	10 248	1.6 %	-
r4_h4_b	10 107	9 613	10 107	4.9 %	-
r4_h4_c	9 641	9 635	9 560	0.1 %	0.8 %
r4_h5b_a	5 399	5 396	10 248	-	-
r4_h5b_b	5 345	5 345	5 345	-	-
r4_h5b_c	5 403	5 403	5 403	-	-
r4_h5c_a	28 600	28 600	25 739	-	10.0 %
r4_h5c_b	25 920	25 920	24 968	-	3.7 %
r4_h5c_c	23 816	23 816	23 816	-	-

Table 9.3: Results for test instances in the set *r4*; objective values for benchmark runs, normal runs with a larger max time, and improvement runs on the benchmark solutions. *The improvement percentage is compared to the benchmark run.

Applied to the fix and relax solutions The improvement heuristic applied to the fix and relax solutions give, for all of the runs, a significant improvement of the objective value. On average, for all the test instances in the sets *r4* and *r3*, the heuristic improves the fix and relax solutions with 5.8%. However, the initial fix and relax solutions were on average poorer than the benchmark solutions. Looking at both the improved benchmark solutions and the improved fix and relax solutions, it seems like the final objective function values of the two methods lie a lot closer to each other than the initial objective function values of the two methods.

Applied to the benchmark solutions from alteration d Overall, the best solutions found for alterations a and b were achieved from applying the improvement heuristic with initial variable values from the benchmark solutions of alteration d. The objective values found with this improvement method can be seen as crosses in the Figures 9.2-9.6. These crosses lie low to the left in all of the graphs, demonstrating decent performance. The total computational time; time for running alteration d added to time for running the improvement algorithm, is lower than the benchmark max time, and the heuristic gives for the most time a better or an as good solution as the benchmark values. This method does, however, not seem to work for neither alteration a nor b for the most complex test instance series, *h4* and *h5c*, for both of the test instance sets. Also, the method is less consistent in finding improved solutions when it is run for 400 seconds compared to 1200 seconds.

For the alterations a, the improvement heuristic using solutions from alteration d succeeded in improving the solutions of alteration a in 20% of the runs. For the improved runs, the heuristic improved the solution with, on average, 2.9%. For the alterations b, the improvement heuristic applied to the alteration d solutions gave improvement in 50% of the runs, improving the solutions, on average, 0.2%.

9.3 Valuation of the recourse model solutions

Different tools to get a quantitative measure of the value of implementing a stochastic model rather than a deterministic model were presented in Section 4.2.1. In this section, the VSS_t and $EVPI$ will be calculated for two test instances, *r4-v_h2* and *r4-v_h3*, generated as described in Chapter 8; one with horizon 2 days and another with horizon 3 days. The values are calculated for runs on alteration b. The reason for choosing this alteration to be tested is that alteration a and c are hard to solve to optimality within reasonable time, and that alteration b seems to performs the best among the stochastic model alterations.

The tests done in this section are therefore done to compare the results obtained deterministically with the results obtained with the stochastic multi-stage model, allowing recourse decisions of type 1 made in every stage - the possibility to postpone a scheduled elective patient one time period.

The deterministic problems were solved using the same model as were written for the stochastic problems. The data input was formatted to suit a node formulation with only one scenario. The different data parameters were calculated as an average of the values in the node formulation, depending on the probabilities for the scenarios to occur. For the parameters defined as integers, the average values were rounded to the closest integer.

9.3.1 Value of stochastic solution

Tables 9.4 and 9.5 present the calculations made to get the series of VSS_t for the two test instances, calculated with equation (4.9), given in Section 4.2.1. The first column of numbers gives the solution to the deterministic problem, the solution to the *expected value problem (EV)*. The running times to solve this problem were small for both instances, as expected.

The next column gives the solution to the corresponding stochastic *recourse problem* (*RP*). For the instance *r4-v_h3* of horizon 3, the solution process was stopped after 12 000 seconds, resulting in a gap between the best bound and best solution of 1.55%. For the test instance *r4-v_h2* of horizon 2, the deterministic objective values are better than the stochastic solutions. But the deterministic model does not account for uncertainties, and the values EEV_t and VSS_t are introduced to see how the deterministic solution works in an uncertain environment.

EEV_1 is defined as the RP solution, where no decision variables are initially fixed. The following columns in the table show the values for the other EEV_t. In each EEV_t, the variables $x_{it'r}, t' = 1...(t-1)$ are fixed from the EV solution. These values illustrate the consequences of applying parts of the EV solution in an uncertain environment. The total expected costs in EEV_t increase for larger t, this is because more variables are initially fixed, before the uncertainties are revealed. The values VSS_t also increase with larger t. This sequence of values represents the cost of ignoring uncertainty until stage t in the decision making. The size of the values VSS_t and the fact that they increase with every t give an indication that it is beneficial to reduce the uncertainty by using a stochastic model, allowing for recourse decisions of type 1. This can also be seen in the lower row, providing the VSS_t/EEV_t relationship. This percentage value indicates the relative improvement in the total costs if the stochastic programming approach is used instead of fixing an amount of variables in the expected value approach.

The best schedule found for test instance *r4-v_h2* with the stochastic multi-stage model performs on average 7.6% better than if the full patient schedule found in the corresponding expected value problem is used. For test instance *r4-v_h3*, the relative improvement using the stochastic multi-stage solution rather than the expected value solution is 4.1%.

In the tables, the costs can be studied on a more detailed level. When many of the variables initially are fixed from the EV, more costs are expected to be spent on postponing elective patients to the next planning horizon, recourse decisions and overtime. The rows underneath the *total expected costs* represent how much higher or lower the different parts of the total costs are, when a larger number of variables initially is fixed compared to having none fixed.

	Det/ EV	*Stoch/* EEV_1	EEV_2	EEV_3	VSS_2	VSS_3
Total running time:	1	140	13	< 1		
Total expected costs:	6 139	6 333	6 675	6 856	**342**	**523**
Best bound:	6 139	6 313	6 669	6 856	356	542
Expected costs of..						
..performing the elective cases:	5 459	5 520	5 320	5 027	-200	-493
..postponing elective cases to the next planning horizon:	630	620	973	1 541	354	921
..postponing the elective cases one time period (recourse 1):	0	90	131	143	41	53
..changing the operating rooms of the elective cases (recourse 2):	0	0	0	0	0	0
..total overtime:	0	26	149	83	123	57
..exceeding the pre-op capacity:	0	8	38	28	30	20
..exceeding the ICU capacity:	0	0	0	0	0	0
..exceeding the PACU capacity:	50	70	65	35	-5	-35
				VSS_t/EEV_t	*5.1 %*	*7.6 %*

Table 9.4: Value of stochastic solution, calculated for test instance *r4-v_h2*, of horizon 2 and alteration *b*

	$Det/$ EV	$Stoch/$ EEV_1	EEV_2	EEV_3	EEV_4	VSS_2	VSS_3	VSS_4
Total running time:	12	12 001	12 000	2 405	< 1			
Total expected costs:	10 013	9 744	9 873	10 005	10 165	**129**	**262**	**421**
Best bound:	10 006	9 592	9 790	9 995	10 155	198	403	563
Expected costs of..								
..performing the elective cases:	7 729	8 320	8 316	7 958	7 526	-4	-362	-793
..postponing elective cases to the next planning horizon:	1 984	947	966	1 564	2 359	19	618	1 412
..postponing the elective cases one time period (recourse 1):	0	152	152	137	72	0	-15	-80
..changing the operating rooms of the elective cases (recourse 2):	0	0	0	0	0	0	0	0
..total overtime:	90	131	147	94	48	15	-37	-83
..exceeding the pre-op capacity:	0	29	59	56	0	30	28	-29
..exceeding the ICU capacity:	0	8	18	13	5	10	5	-3
..exceeding the PACU capacity:	210	158	216	183	155	59	25	-3
					VSS_t/EEV_t	*1.3 %*	*2.6 %*	*4.1 %*

Table 9.5: Value of stochastic solution, calculated for test instance *r4-v_h3*, of horizon 3 and alteration *b*

9.3.2 Expected value of perfect information

Tables 9.6 and 9.7 present the calculations made to obtain the $EVPI$ values for the two test instances. The first column of the tables present the *expected value problem (EV)* solution, the solution to the deterministic problem. The next columns present the *Wait-and-see solutions (WS)* for the different scenarios, The WS values are the expected objective values of the best solutions to every scenario, if no uncertainties are present and all information is known. The column *Avg.* WS is the average of the WS solutions, and the RP column gives the stochastic solution. The $EVPI$, calculated with equation (4.6), given in Section 4.2.1, is a measure of how much it is worth to eliminate all the uncertainties in the stochastic model.

For both of the problems, it is clear that it is beneficial to reduce the uncertainties in the model. Especially for the test instance *r4-v_h3* of horizon 3, the $EVPI$ is large. Studying the WS-solutions, one can see that the total expected costs for scenario 1-4 are substantially higher than for scenario 5-8. This is because a lot more emergency patients arrive in $t = 1$ for the scenarios 1-4 than in scenarios 4-8, 7 patients compared to 3. The potential from removing the uncertainties regarding the emergency patient arrivals in $t = 1$ is therefore of large value. The stochastic model must make a schedule including the possibility that both few and many emergency patients may arrive in $t = 1$. The VSS_t is therefore small compared to the $EVPI$.

For the test instance *r4-v_h2* of horizon 2, the $EVPI$ is smaller than the VSS_t values. There is little difference between the amounts of emergency patient arriving in the different scenarios. In $t = 1$ there are 4 emergency patients arriving in scenarios 1-4 compared to 3 emergency patients arriving in scenarios 5-8. The total expected costs are more even between the WS solutions in this test instance than in test instance *r4-v_h3*.

	Det/ EV	*Wait-and-see solutions*				*Avg. WS*	*Stoch/ RP*	*EVPI*
		Sc1	*Sc2*	*Sc3*	*Sc4*			
probability		0.25	0.25	0.25	0.25			
Total running time:	< 1	1	< 1	< 1	< 1	< 1	140	139
Total expected costs:	6 139	5 893	5 977	6 198	6 310	6 094	6 333	**239**
Best bound:	6 139	5 893	5 949	6 187	6 305	6 083	6 313	230
Expected costs of..								
..performing the elective cases:	5 459	5 843	5 769	5 445	5 255	5 578	5 520	-58
..postponing elective cases to the next planning horizon:	630	0	150	633	947	433	620	187
..postponing the elective cases one time period (recourse 1):	0	0	0	0	0	0	90	90
..changing the operating rooms of the elective cases (recourse 2):	0	0	0	0	0	0	0	0
..total overtime:	0	0	18	0	18	9	26	18
..exceeding the pre-op capacity:	0	0	0	0	0	0	8	8
..exceeding the ICU capacity:	0	0	0	0	0	0	0	0
..exceeding the PACU capacity:	50	50	40	120	90	75	70	-5

Table 9.6: Expected value of perfect information, calculated for test instance *r4-v_h2* of horizon 2, alteration *b*

	Det/ EV	*Wait-and-see solutions* *Sc1*	*Sc2*	*Sc3*	*Sc4*	*Sc5*	*Sc6*	*Sc7*	*Sc8*	*Avg. WS*	*Stoch/ RP*	*EVPI*
probability		0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125			
Total running time:	12	8	9	9	3	5	6	8	6	7	12 001	11 994
Total expected costs:	10 013	9 179	9 044	9 054	9 574	5 655	5 694	6 004	5 717	7 490	9 744	**2 254**
Best bound:	10 006	9 148	9 036	9 011	9 550	5 637	5 670	5 978	5 699	7 466	9 592	2 126
Expected costs of..												
..performing the elective cases:	7 729	8 388	8 542	8 789	7 813	4 795	4 784	4 493	4 777	6 548	8 320	1 772
..postponing elective cases to the next planning horizon:	1 984	623	312	0	1 603	0	0	481	0	377	947	569
..postponing the elective cases one time period (recourse 1):	0	0	0	0	0	572	572	572	572	286	152	-134
..changing the operating rooms of the elective cases (recourse 2):	0	0	0	0	0	0	0	0	0	0	0	0
..total overtime:	90	18	0	35	18	18	18	18	18	18	131	114
..exceeding the pre-op capacity:	0	0	10	0	0	130	130	130	130	66	29	-38
..exceeding the ICU capacity:	0	0	0	0	0	0	0	100	60	20	8	-13
..exceeding the PACU capacity:	210	150	180	230	140	140	190	210	160	175	158	-18

Table 9.7: Expected value of perfect information, calculated for test instance *r4-v_h3* of horizon 3, alteration *b*

9.4 Valuation of including the pre-op and post-op capacity limits

To get a quantitative measure of the value of including the pre-op and post-op capacity limits in the ORSP, 10 test instances in the set *r4-r* of horizon 2 were run for alteration b in two variations; with and without costs for exceeding pre-op and post-op capacities, denoted variation A and B. By removing these costs, the capacity limits of pre-op and post-op are disregarded. The size of horizon is chosen small to make sure that the solution values found have small gaps to the best bounds found.

The solutions found for variation A, disregarding the pre-op and post-op capacity constraints, were used as fixed decision variables in variation B, to find the total expected costs if the capacity constraints are neglected in the planning phase. The average results for the runs of the 10 instances are shown in Table 9.8.

The value of including the pre-op and post-op capacity limits in the planning phase is shown in column D. For the test instances in the set *r4-r*, there is a small benefit of considering the capacity constraints of pre-op and post-op in the planning phase, variation C. With parameter values set as described in Section 8.2, the expected costs can be reduced by 0.2%. This is not a significant improvement, but the value may change if other capacities are chosen for the pre-op and post-op facilities, the costs are changed, or the number of emergency patients arriving is changed. This is not studied further in this thesis.

	A *Capacity limits* dis-regarded*	B *Capacity limits* included*	C *Capacity limits* included, fixed values from* A	$D = (C - B)$ *Value of including pre-op and post-op constraints*
Total running time:	1943	2348	< 1	
Total expected costs:	6 304	6 386	6 396	**11**
Best bound:	6 296	6 371	6 395	24
Gap:	0.14 %	0.22 %	0.02 %	
Expected costs of..				
..performing the elective cases:	5 558	5 480	5 524	45
..postponing elective cases to the next planning horizon:	582	714	638	-76
..postponing the elective cases one time period (recourse 1):	87	83	93	10
..changing the operating rooms of the elective cases (recourse 2):	0	0	0	0
..total overtime:	77	41	55	14
..exceeding the pre-op capacity:	0	5	12	7
..exceeding the ICU capacity:	0	16	20	5
..exceeding the PACU capacity:	0	49	55	7

Table 9.8: Value of pre-op and post-op capacity constraints, calculated for an average of 10 test instances in the set *r4-r*, of horizon 2 and alteration *b*. *The capacity limits of the pre-op and post-op facilities.

9.5 Testing the value of reserving an OR for the emergency patients only

The model presented in this thesis can be used to see if it is reasonable to reserve an OR for emergency patients only. If this is the case, the scheduling problem would be less complicated to solve, as fewer emergency patients would be expected to interfere the capacity intended for the elective patients, making it easier to make a surgery schedule. 10 test instances in the set *r4-r* of horizon 2 were run for alteration *b* and two variations: reserving and not reserving an OR for the emergency patients. The size of horizon is chosen small to make sure that the solution values found have small gaps to the best bounds found. On average for the 10 test instances, the gap after the max time of 3600 seconds was reached was 0.67% for the runs with 1 OR reserved, and 0.22% for the runs with no ORs reserved.

Figure 9.7 shows the results of the runs. For the instances tested on the ORSP of alteration *b* it is on average 2.8% more costly to reserve a full OR for emergency patients, than sharing the capacity of all ORs between the two patient groups. The filled circles show the solutions found when one of the four ORs is reserved for emergency patients only (the emergency patients can also go to one of the other ORs), and the non-filled circles show the solutions found when the set of ORs that the patients can go to is as described in Section 8.2, and the capacity is shared between the two types of patients.

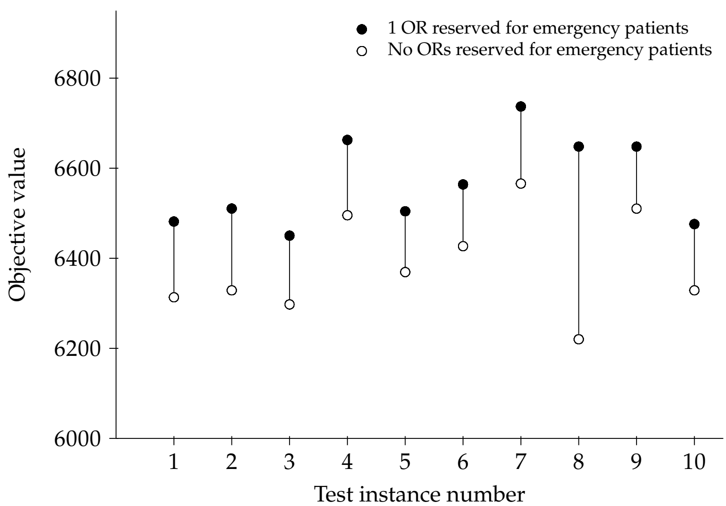


Figure 9.7: Results for test instances in the set *r4-r* for horizon 2, objective values with 1 of 4 ORs reserved for emergency patients, compared to no ORs reserved.

10. Conclusions

This thesis has defined and described the operating room scheduling problem (ORSP). The problem consists of allocating elective cases to time periods and operating rooms, considering the uncertainty regarding emergency patient arrivals and the capacity constraints of the pre-op and the post-op facilities.

Two mathematical models have been presented; A deterministic model and a multi-stage stochastic model, taking the uncertainty of the arrivals of non-elective patients into account. The stochastic model is presented in both a scenario formulation and a node formulation. It allows for two reactions in every stage after the uncertain information regarding the emergency patients has been revealed: the originally scheduled elective patients may be postponed one time period (type 1) or changed operating room for within the same time period (type 2). In addition, the arriving emergency patients must be allocated to ORs. The models presented provide a tool that can support the decision making regarding the scheduling of the elective patients in a medium to short term perspective.

The node formulation of the stochastic model was implemented in Xpress^{MP}. To test the model, several fictive data instances were generated to simulate real-life data for a medium-sized hospital in Norway. The instances include two possible numbers of emergency patient arrivals in each time period. The results show that the implemented problems that allow recourse decisions of type 2 soon become too complex, giving solutions far from the best bounds. Allowing recourse decisions of type 1 give good results for the test instances of a smaller size, evaluating 4-8 scenarios in total.

Two heuristics have been applied to the model: The constructive heuristic "fix and relax" and an improvement heuristic. The fix and relax heuristic provides medium good solutions, with the potential to be improved with the improvement algorithm. The improvement algorithm is beneficial to use for the smaller instances, but does not find better solutions for the most complex test instances with horizons of 4 and 5 time periods.

For the smaller instances, the method that performed the best, in terms of solution quality and time, was the improvement heuristic with initial variable values found in a run *not allowing* the defined recourse decisions, on a problem allowing recourse decisions of type 1. For the most complex instances, the gaps between the best solutions found and the best bounds were large for the problems allowing recourse decisions of type 1 or 2, for all methods. The best solutions for these instances were obtained when neither allowing type 1 nor type 2 decisions.

The results indicate that it is advantageous to include the uncertainty using a multi-stage model, allowing recourse decisions of type 1. Using the solution approaches presented, it may be valuable to try to obtain realistic approximations of the uncertain parameters in the later time periods, rather than include them as uncertain in the model, to avoid too complex problems. Including the pre-op and post-op capacity constraints in the ORSP seem to be of a small benefit for the test instances.

11. Further work

In this chapter, the areas considered most relevant for further investigation or development are presented.

Real data

As the real data becomes easier available from the *DIPS* system [22], it will be natural to test the model using actual values, both historical and present. Using historical data, where all information is known and no uncertainties are present, it is possible to run the model for the information known at a specific point of time, and see how the solution would work out in the scenario that actually evolved. This solution could also be compared to the schedule that actually was decided on.

Easing the way of treating the data input

If the model is to be used repeatedly and on larger problems, the way of formatting the data input should be more intuitive and uncomplicated. Currently, the data input is manually organized in a spreadsheet in MS Excel, then copied into a *.txt*-file, handled by Xpress$^{\text{MP}}$. The model would be easier to use if it automatically connected to a user friendly data input interface/application.

Include more details in the model

The model presented in this thesis is, like most optimization models, a simplification of the reality. There are a number of extensions that can be made, but it will not always be possible as a result of limited computational capacity. Below follows a few of the possible extensions that can be considered.

Add an urgency level to the emergency cases In the model presented in this thesis, the emergency patients are all classified under the same level of urgency; they have to be treated in the time period of arrival. In reality, there is more than one urgency level [22], and to simulate the reality in a better way, this should be included in the model.

Extension of the recourse decisions of type 1 The recourse decisions of type 1 only allow for postponing the elective patients *one* time period. If the reason for postponing the patients is that there is a bottleneck in ICU or PACU, one would possibly want to postpone an elective patient further. This would, however, lead to a lot more complex model, and the possible benefits of changing the rules need to be weighted against the computational cost.

Including the opportunity to give elective cases different priorities Some of the elective cases may be more urgent than others. Also, some elective cases may have been postponed from the last scheduling horizon, and should be guaranteed to be treated in the current scheduling horizon.

Getting into a more detailed level of the resources In the problem described in this thesis, each operating room corresponds to a given set of resources. It could be beneficial to split these "resource groups" into for example surgeons, nurses, equipment and rooms. This could enable the model to for example communicate with other models, such as nurse rostering or surgeon’s working schedules, and it would give a better representation of the reality.

A. Results from the computational study

Variation	Method	# It.	Max time	Running time	Best bound	Obj. value	Gap	Impro-vement
r3_h2_a	Benchmark		3 600	3 600	4 271	7 233	40.9 %	
	Improved	3	1 200	1 201		nbs	-	0.0 %
	Improved, values from *d*	3	400	400		7 068	39.6 %	
	Fix-and-relax	3	1 200	701		7 104	39.9 %	
	Improved	3	1 200	1 201		7 066	39.6 %	0.5 %
r3_h2_b	Benchmark		3 600	9	7 027	7 055	0.4 %	
	Improved	3	1 200	1 200		7 041	0.2 %	0.2 %
	Improved, values from *d*	3	400	394		7 057	0.4 %	
	Fix-and-relax	3	1 200	474		7 043	0.2 %	
	Improved	3	1 200	1 201		7 042	0.2 %	0.0 %
r3_h2_c	Benchmark		3 600	3 600	4 126	7 471	44.8 %	
	Improved	3	1 200	1 213		7 344	43.8 %	1.7 %
r3_h2_d	Benchmark		3 600	< 1	7 209	7 238	0.4 %	
r3_h3_a	Benchmark		3 600	3 601	6 083	10 947	44.4 %	
	Improved	3	1 200	1 202		nbs	-	0.0 %
	Improved, values from *d*	3	400	400		nbs	-	
	Fix-and-relax	4	1 200	752		10 808	43.7 %	
	Improved	3	1 200	1 201		10 696	43.1 %	1.0 %
r3_h3_b	Benchmark		3 600	3 600	10 361	10 455	0.9 %	
	Improved	3	1 200	1 200		nbs	-	0.0 %
	Improved, values from *d*	3	400	400		10 475	1.1 %	
	Fix-and-relax	4	1 200	753		10 696	3.1 %	
	Improved	3	1 200	1 201		nbs	-	0.0 %
r3_h3_c	Benchmark		3 600	3 601	5 573	11 291	50.6 %	
	Improved	3	1 200	1 213		nbs	-	0.0 %
r3_h3_d	Benchmark		3 600	< 1	10 765	10 817	0.5 %	
r3_h4_a	Benchmark		3 600	11 162	7 033	16 844	58.2 %	
	Improved	3	1 200	1 200		nbs	-	0.0 %
	Improved, values from *d*	3	400	400		nbs	-	
	Fix-and-relax	5	1 200	792		19 634	64.2 %	
	Improved	3	1 200	4 007		16 520	57.4 %	15.9 %
r3_h4_b	Benchmark		3 600	3 602	11 223	17 339	35.3 %	
	Improved	3	1 200	1 201		nbs	-	0.0 %
	Improved, values from *d*	3	400	400		nbs	-	
	Fix-and-relax	5	1 200	802		19 193	41.5 %	
	Improved	3	1 200	1 204		15 728	28.6 %	18.1 %
r3_h4_c	Benchmark		3 600	3 601	6 748	14 717	54.2 %	
	Improved	3	1 200	1 201		nbs	-	0.0 %
r3_h4_d	Benchmark		3 600	1	14 644	14 717	0.5 %	

Table A.1: Overview of results for test instances *r*3 (1/2)

Variation	Method	# It.	Max time	Running time	Best bound	Obj. value	Gap	Impro-vement
r3_h5b_a	Benchmark		3 600	3 601	2 378	5 209	54.4 %	
	Improved	3	1 200	1 203		3 682	35.4 %	29.3 %
	Improved, values from *d*	3	400	521		nbs	-	
	Improved, values from *d*	5	400	729		nbs	-	
	Fix-and-relax	6	1 200	205		5 924	59.9 %	
	Improved	3	1 200	1 201		5 275	54.9 %	10.9 %
r3_h5b_b	Benchmark	3	3 600	3 600	4 930	5 057	2.5 %	
	Improved	3	1 200	1 202		5 054	2.5 %	0.1 %
	Improved, values from *d*	3	400	401		5 058	2.5 %	
	Improved, values from *d*	5	400	403		5 099	3.3 %	
	Fix-and-relax	6	1 200	183		5 442	9.4 %	
	Improved	3	1 200	1 201		5 055	2.5 %	7.1 %
r3_h5b_c	Benchmark	3	3 600	3 601	4 519	5 159	12.4 %	
	Improved	3	1 200	1 201		nbs	-	0.0 %
r3_h5b_d	Benchmark		3 600	< 1	5 197	5 204	0.1 %	
r3_h5c_a	Benchmark		3 600	25 731	8 250	26 480	68.8 %	
	Improved	3	1 200	37 010		nbs	-	0.0 %
	Improved, values from *d*	3	400	18 269		nbs	-	
	Improved, values from *d*	5	400	24 211		nbs	-	
	Fix-and-relax	6	1 200	14 448		27 401	69.9 %	
	Improved	3	1 200	32 136		26 262	68.6 %	4.2 %
r3_h5c_b	Benchmark	3	3 600	3 603	11 337	25 610	55.7 %	
	Improved	3	1 200	1 230		nbs	-	0.0 %
	Improved, values from *d*	3	400	400		nbs	-	
	Improved, values from *d*	5	400	402		nbs	-	
	Fix-and-relax	6	1 200	869		26 008	56.4 %	
	Improved	3	1 200	1 213		26 001	56.4 %	0.0 %
r3_h5c_c	Benchmark		3 600	3 601	12 587	24 217	48.0 %	
	Improved	3	1 200	1 203		nbs	-	0.0 %
r3_h5c_d	Benchmark		3 600	1	23 815	23 913	0.4 %	

Table A.2: Overview of results for test instances *r*3 (2/2)

Variation	Method	# It.	Max time	Running time	Best bound	Obj. value	Gap	Improvement
r4_h2_a	Benchmark		3 600	3 600	3 750	6 715	44.2 %	
	Improved	3	1 200	1 202		6 711	44.1 %	0.1 %
	Improved, values from *d*	3	400	400		6 509	42.4 %	
	Improved, values from *d*	3	1 200	1 201		6 474	42.1 %	
	Benchmark		4 800	4 800	3 761	6 699	43.9 %	
	Fix-and-relax	3	1 200	701		6 541	42.7 %	
	Improved	3	1 200	1 201		6 484	42.2 %	0.9 %
r4_h2_b	Benchmark		3 600	121	6 407	6 440	0.5 %	
	Improved, values from *d*	3	400	399		6 446	0.6 %	
	Improved, values from *d*	3	1 200	1 201		6 442	0.5 %	
	Fix-and-relax	3	1 200	701		6 443	0.6 %	
	Improved	3	1 200	1 201		6 435	0.4 %	0.1 %
r4_h2_c	Benchmark		3 600	3 600	4 324	6 609	34.6 %	
	Improved	3	1 200	1 201		6 601	34.5 %	0.1 %
	Benchmark		4 800	4 800	4 340	6 609	34.3 %	
r4_h2_d	Benchmark		3 600	< 1	6 518	6 535	0.3 %	
r4_h3_a	Benchmark		3 600	3 601	4 568	10 493	56.5 %	
	Improved	3	1 200	1 203		10 215	55.3 %	2.7 %
	Improved, values from *d*	3	400	402		nbs	-	
	Improved, values from *d*	3	1 200	1 201		nbs	-	
	Benchmark		4 800	4 801	4 589	10 493	56.3 %	
	Fix-and-relax	4	1 200	427		10 650	57.1 %	
	Improved	3	1 200	1 201		10 248	55.4 %	3.8 %
r4_h3_b	Benchmark		3 600	3 600	9 595	9 954	3.6 %	
	Improved	3	1 200	1 201		9 916	3.2 %	0.4 %
	Improved, values from *d*	3	400	400		10 035	4.4 %	
	Improved, values from *d*	3	1 200	1 201		9 909	3.2 %	
	Benchmark		4 800	4 800	9 604	9 945	3.4 %	
	Fix-and-relax	4	1 200	753		10 045	4.5 %	
	Improved	3	1 200	1 201		9 909	3.2 %	1.4 %
r4_h3_c	Benchmark		3 600	3 600	4 274	10 532	59.4 %	
	Improved	3	1 200	1 201		10 132	57.8 %	3.8 %
	Benchmark		4800	4800	4 295	10 532	59.2 %	
r4_h3_d	Benchmark		3 600	16	10 085	10 134	0.5 %	
r4_h4_a	Benchmark		3 600	3 603	4 124	10 248	59.8 %	
	Improved	3	1 200	5 007		nbs	-	0.0 %
	Improved, values from *d*	3	400	410		nbs	-	
	Improved, values from *d*	3	1 200	1 210		nbs	-	
	Benchmark		4 800	4 800	4 124	10 086	59.1 %	
	Fix-and-relax	5	1 200	791		11 831	65.1 %	
	Improved	3	1 200	1 208		11 523	64.2 %	2.6 %
r4_h4_b	Benchmark		3 600	3 602	8 191	10 107	19.0 %	
	Improved	3	1 200	1 213		nbs	-	0.0 %
	Improved, values from *d*	3	400	400		nbs	-	
	Improved, values from *d*	3	1 200	1 211		nbs	-	
	Benchmark		4 800	4 800	8 193	9 613	14.8 %	
	Fix-and-relax	5	1 200	790		12 496	34.4 %	
	Improved	3	1 200	1 208		10 700	23.4 %	14.4 %
r4_h4_c	Benchmark		3 600	3 601	3 391	9 641	64.8 %	
	Improved	3	1 200	1 202		9 560	64.5 %	0.8 %
	Benchmark		4 800	4 801	3 402	9 635	64.7 %	
r4_h4_d	Benchmark		3 600	2	9 475	9 513	0.4 %	

Table A.3: Overview of results for test instances *r*4 (1/2)

Variation	Method	# It.	Max time	Running time	Best bound	Obj. value	Gap	Impro-vement
r4_h5b_a	Benchmark		3 600	3 601	3 561	5 399	34.0 %	
	Improved	3	1 200	1 201		nbs	-	0.0 %
	Improved, values from *d*	3	400	458		nbs	-	
	Improved, values from *d*	5	400	577		nbs	-	
	Improved, values from *d*	3	1 200	1 202		5 427	34.4 %	
	Improved, values from *d*	5	1 200	1 206		5 426	34.4 %	
	Benchmark		4 800	4 801	3 567	5 396	33.9 %	
	Fix-and-relax	6	1 200	185		5 845	39.1 %	
	Improved	3	1 200	1 203		5 398	34.0 %	7.7 %
r4_h5b_b	Benchmark		3 600	3 601	5 118	5 345	4.2 %	
	Improved	3	1 200	1 201		5 345	4.2 %	0.0 %
	Improved, values from *d*	3	400	400		5 351	4.3 %	
	Improved, values from *d*	5	400	403		5 352	4.4 %	
	Improved, values from *d*	3	1 200	1 202		5 344	4.2 %	
	Improved, values from *d*	5	1 200	1 202		5 345	4.2 %	
	Benchmark		4 800	4 801	5 128	5 345	4.1 %	
	Fix-and-relax	6	1 200	184		5 745	10.9 %	
	Improved	3	1 200	1 201		5 345	4.2 %	7.0 %
r4_h5b_c	Benchmark		3 600	3 600	4 843	5 403	10.4 %	
	Improved	3	1 200	1 204		nbs	-	0.0 %
	Benchmark		4 800	4 800	4 848	5 403	10.3 %	
r4_h5b_d	Benchmark		3 600	< 1	5 443	5 444	0.0 %	
r4_h5c_a	Benchmark		3 600	15 972	6 028	28 600	78.9 %	
	Improved	3	1 200	17 484		25 739	76.6 %	10.0 %
	Improved, values from *d*	3	400	11 552		nbs	-	
	Improved, values from *d*	5	400	415		nbs	-	
	Improved, values from *d*	3	1 200	11 768		nbs	-	
	Improved, values from *d*	5	1 200	1 238		nbs	-	
	Benchmark		4 800	16 005	6 028	28 600	78.9 %	
	Fix-and-relax	6	1 200	5 247		27 206	77.8 %	
	Improved	3	1 200	12 208		25 830	76.7 %	5.1 %
r4_h5c_b	Benchmark		3 600	3 604	9 798	25 920	62.2 %	
	Improved	3	1 200	1 224		24 968	60.8 %	3.7 %
	Improved, values from *d*	3	400	400		nbs	-	
	Improved, values from *d*	5	400	402		nbs	-	
	Improved, values from *d*	3	1 200	1 210		nbs	-	
	Improved, values from *d*	5	1 200	1 213		nbs	-	
	Benchmark		4 800	4 804	9 798	25 920	62.2 %	
	Fix-and-relax	6	1 200	876		30 474	67.8 %	
	Improved	3	1 200	1 215		24 859	60.6 %	18.4 %
r4_h5c_c	Benchmark		3 600	3 601	8 149	23 816	65.8 %	
	Improved	3	1 200	1 204		nbs	-	0.0 %
	Benchmark		4 800	4 801	8 155	23 816	65.8 %	
r4_h5c_d	Benchmark		3 600	2	23 253	23 360	0.5 %	

Table A.4: Overview of results for test instances *r*4 (2/2)

B. Mosel code

This is the Mosel code written for normal runs of the ORSP. The code file can be found at the enclosed CD, together with the Mosel codes for the fix and relax heuristic and the improvement heuristic.

```
! Model for the Operation Room Scheduling Problem (ORSP)
! Multi-stage - Node formulation
! Created by Sofie Døving Agdestein
! 2012-05-30

model OR_scheduling
options explterm
options noimplicit
uses "mmxprs", "mmsystem";

!-----------------------------------------------------------------------------------------------------------------------
! UPDATE PARAMETERS FOR EACH RUN
parameters
    DataFile ='120509_h2_01_a.txt';                    !READ: Original data file
    DataFileFix = 'Output/120509_h2_01_a_FIX.txt';     !WRITE: File to write out best values (for improvement code)
    Output = 'Output/120509_h2_01_a_Output.txt';       !WRITE: File to write output to
    Comments = 'Tuned, max time 3600 sec. V60';
    Maxtime = 3600;
end-parameters
!-----------------------------------------------------------------------------------------------------------------------

declarations
    nPeriod:            integer; !Horizon
    nElective:          integer;
    nEmergency:         integer;
    nRooms:             integer;
    nNode:              integer;
    nNode_Horizon:      integer;
    nPriceRange:        integer;
    starttime:          real;
end-declarations

starttime := gettime;

initializations from DataFile
    nPeriod;
    nElective;
    nEmergency;
    nRooms;
    nNode;
    nNode_Horizon;
    nPriceRange;
end-initializations

declarations
    Horizon             = 1 .. nPeriod;
    Horizon_C           = 1 .. (nPeriod+1);
    Elective            = 1 .. nElective;
    Emergency           = 1 .. nEmergency;
    Rooms               = 1 .. nRooms;
    Node                = 2 .. nNode;
    Node_Horizon        = 2 .. nNode_Horizon;
    PriceRange          = 1 .. nPriceRange;
end-declarations

finalize(Horizon);
finalize(Horizon_C);
finalize(Elective);
finalize(Emergency);
finalize(Rooms);
finalize(Node);
finalize(Node_Horizon);
finalize(PriceRange);

declarations
    Probability_Node:       array(Node)                         of real;        !Probability of visiting node nn (from  its parent node)
    Time_Node:              array(Node)                         of integer;     !Time period of node nn = t(nn)
```

```
    Preceding_Node:         array(Node)                         of integer;     !Preceding node of node nn
    Elective_PostOp:        array(Elective)                     of string;      !Where the Elective patients ii are going for post-op
    Emergency_Node:         array(Emergency)                    of integer;     !Which time period t(nn) the Emergency patients jj are arriving
    Emergency_t_ICU:        array(Node_Horizon)                 of integer;     !# of Em. patients at ICU in time period t(nn) in the sc. passing nn
    Emergency_t_PACU:       array(Node_Horizon)                 of integer;     !# of Em. patients at PACU in time period t(nn) in the sc. passing nn
    Rooms_Elective_set:     array(Elective)                     of set of integer; !The set of rooms rr that each Elective case ii can go to
    Rooms_Emergency_set:    array(Emergency, Node_Horizon)      of set of integer; !Which rooms rr each Em. case jj in the sc. passing nn can go to
    Release_Time:           array(Elective)                     of integer;     !The earliest time period tt for performing Elective case ii
    Cost_Elective:          array(Elective, Rooms, Node)        of real;        !Cost of Elective case ii in OR rr, time t(nn) and in the sc. passing nn
    Cost_Recourse_1:        array(Elective, Horizon, Rooms)     of integer;     !Cost of postponing Elective case ii one time p, to time tt+1 and OR rr
    Cost_Recourse_2:        array(Elective, Horizon, Rooms)     of integer;     !Cost of moving elective case ii to OR rr in time period tt
    Cost_Overtime:          array(Horizon, Rooms, PriceRange)   of integer;     !Cost per unit of overtime of price range pp in room rr and time p. t(nn)
    Cost_PREOP:             array(Horizon)                      of integer;     !Cost per patient exceeding the capacity of the pre-op in time period tt
    Cost_ICU:               array(Horizon)                      of integer;     !Cost per patient exceeding the capacity of ICU in time period tt
    Cost_PACU:              array(Horizon)                      of integer;     !Cost per patient exceeding the capacity of PACU in time period tt
    Time_Elective:          array(Elective, Node_Horizon)       of real;        !Time needed for performing Elective case ii in the sc. passing nn
    Time_Emergency:         array(Emergency, Node_Horizon)      of real;        !Time needed for performing Emergency case jj in the sc. passing nn
    Regular_Capacity:       array(Rooms, Node_Horizon)          of real;        !Total regular capacity of OR rr in time t(nn) and the sc. passing nn
    PriceRange_Capacity:    array(Rooms, Node_Horizon, PriceRange) of real;     !Overtime cap. of price pp of OR rr in time t(nn), for the sc. passing nn
    Capacity_PREOP:         array(Horizon)                      of integer;     !Available capacity of pre-op in time period tt
    Capacity_ICU:           array(Horizon)                      of integer;     !Available capacity of ICU in time period tt
    Capacity_PACU:          array(Horizon)                      of integer;     !Available capacity of PACU in time period tt
    Duration_PO_Elective:   array(Elective, Node_Horizon)       of integer;     !Duration of stay at post-op for El. case ii in the sc. passing nn
end-declarations

initializations from DataFile
    Probability_Node;
    Time_Node;
    Preceding_Node;
    Elective_PostOp;
    Emergency_Node;
!   Emergency_PostOp;
    Emergency_t_ICU;
    Emergency_t_PACU;
    Rooms_Elective_set;
    Rooms_Emergency_set;
    Release_Time;
    Cost_Elective;
    Cost_Recourse_1;
    Cost_Recourse_2;
    Cost_Overtime;
    Cost_PREOP;
    Cost_ICU;
    Cost_PACU;
    Time_Elective;
    Time_Emergency;
    Regular_Capacity;
    PriceRange_Capacity;
    Capacity_PREOP;
    Capacity_ICU;
    Capacity_PACU;
    Duration_PO_Elective;
end-initializations

!--------------------------------------------------------------     CREATING SETS    ------------------------------------------------------------------------

declarations
    Emergency_t_set:            array(Node_Horizon) of set of integer;  !Set of Em. patients jj, arriving in t(nn) in the scenarios passing through node nn
    Emergency_t_amount:         array(Node_Horizon) of integer;         !# of Em. patients jj arriving in t(nn) in the sc. passing through node nn
    Elective_ICU_set:                               set of integer;     !Set of Elective patients ii going to ICU for post-op
    Elective_PACU_set:                              set of integer;     !Set of Elective patients ii going to PACU for post-op
    Earlier_Nodes_set:          array(Node_Horizon) of set of integer;  !Set of nodes preceding node nn, within the horizon
    temp_node:                                      integer;
end-declarations

!Creating set of Elective patients going to ICU/PACU
forall (ii in Elective) do
```

```
    if (Elective_PostOp(ii) = "ICU") then
        Elective_ICU_set += {ii};
    end-if
    if (Elective_PostOp(ii) = "PACU") then
        Elective_PACU_set += {ii};
    end-if
end-do

!Creating sets of Emergency patients, depending on t
forall (jj in Emergency) do
    Emergency_t_set(Emergency_Node(jj)) += {jj};
end-do

!Calculating the size of Emergency_t_set(nn) for every node
forall (nn in Node_Horizon) do
    Emergency_t_amount(nn):=
    getsize(Emergency_t_set(nn));
end-do

!Creating set of earlier nodes within the horizon (not including origin node and end-leaf nodes)
forall (nn in Node_Horizon) do
    Earlier_Nodes_set(nn) += {nn};
    temp_node := Preceding_Node(nn);
    while (temp_node > 1) do
        Earlier_Nodes_set(nn) += {temp_node};
        temp_node := Preceding_Node(temp_node);
    end-do
    finalize(Earlier_Nodes_set(nn));
end-do

!---------------------------------------------------------------     VARIABLES     ---------------------------------------------------------------------------

!Declaring variables
declarations
    binary_elective:                dynamic array(Elective, Horizon_C, Rooms)       of mpvar;
    binary_emergency:               dynamic array(Emergency, Rooms, Node_Horizon)   of mpvar;
    overtime:                       dynamic array(Rooms, Node_Horizon, PriceRange)  of mpvar;
    exceeded_PREOP:                 dynamic array(Node_Horizon)                     of mpvar;
    exceeded_ICU:                   dynamic array(Node_Horizon)                     of mpvar;
    exceeded_PACU:                  dynamic array(Node_Horizon)                     of mpvar;
    binary_recourse_1:              dynamic array(Elective, Rooms, Node_Horizon)    of mpvar;
    binary_recourse_2:              dynamic array(Elective, Rooms, Node_Horizon)    of mpvar;
    binary_actual:                  dynamic array(Elective, Rooms, Node)            of mpvar;
    Allow_Recourse_1:                                                               boolean;
    Allow_Recourse_2:                                                               boolean;
    Fixed_binary_elective:                                                          boolean;
    Value_binary_elective:          dynamic array(Elective, Horizon_C, Rooms)       of integer;
    Constraint_fix_binary_elective: dynamic array(Elective, Horizon_C, Rooms)       of linctr;
end-declarations

initializations from DataFile
    Allow_Recourse_1;
    Allow_Recourse_2;
    Fixed_binary_elective;
    Value_binary_elective;
end-initializations


!Creating the variable binary_elective(ii,rr,nn)
forall (ii in Elective) do
    forall (tt in Horizon_C | tt >= Release_Time(ii)) do
        forall (rr in Rooms_Elective_set(ii)) do
            create(binary_elective(ii,tt,rr));
            binary_elective(ii,tt,rr) is_binary;
        end-do
    end-do
end-do
```

```
!Fixing the binary_elective(ii,tt,rr) variables to the given values Value_binary_elective(ii,tt,rr) if Fixed_binary_elective = true
if (Fixed_binary_elective = true) then
    forall (ii in Elective, tt in Horizon_C, rr in Rooms_Elective_set(ii) | exists(Value_binary_elective(ii,tt,rr)) and tt >= Release_Time(ii)) do
        Constraint_fix_binary_elective(ii,tt,rr) :=
        binary_elective(ii,tt,rr) = Value_binary_elective(ii,tt,rr);
    end-do
end-if

!Creating the variables binary_actual(ii,rr,nn)
forall (ii in Elective) do
    forall (nn in Node | Time_Node(nn) >= Release_Time(ii)) do
        forall (rr in Rooms_Elective_set(ii)) do
            create(binary_actual(ii,rr,nn));
            binary_actual(ii,rr,nn) is_binary;
        end-do
    end-do
end-do

!Creating the variables binary_recourse_1(ii,rr,nn) and binary_recourse_2(ii,rr,nn)
forall (ii in Elective) do
    forall (nn in Node_Horizon | Time_Node(nn) >= Release_Time(ii)) do
        forall (rr in Rooms_Elective_set(ii)) do
            if (Allow_Recourse_1 = true) then
            create(binary_recourse_1(ii,rr,nn));
            binary_recourse_1(ii,rr,nn) is_binary;
            end-if
            if (Allow_Recourse_2 = true) then
            create(binary_recourse_2(ii,rr,nn));
            binary_recourse_2(ii,rr,nn) is_binary;
            end-if
        end-do
    end-do
end-do

!Creating the variable binary_emergency(jj,rr,nn)
forall (nn in Node_Horizon) do
    forall (jj in Emergency_t_set(nn)) do
        forall (rr in Rooms_Emergency_set(jj,nn)) do
            create(binary_emergency(jj,rr,nn));
            binary_emergency(jj,rr,nn) is_binary;
        end-do
    end-do
end-do

!Creating the overtime variables and exceeded capacity variables
forall (nn in Node_Horizon) do
    forall (rr in Rooms) do
        forall (pp in PriceRange) do
            create(overtime(rr,nn,pp));
        end-do
    end-do
    create(exceeded_PREOP(nn));
    create(exceeded_ICU(nn));
    create(exceeded_PACU(nn));
end-do

!--------------------------------------------------------------    OBJECTIVE FUNCTION  ------------------------------------------------------------------------

!Declaring objective function
declarations
    Costs:                                          linctr;
    Cost_Elective_Horizon_Total:                    linctr;
    Cost_Elective_NextHorizon_Total:                linctr;
    Cost_Recourse_1_Total:                          linctr;
    Cost_Recourse_2_Total:                          linctr;
    Cost_Overtime_Total:                            linctr;
```

```
    Cost_Exceeded_PREOP_Total:                      linctr;
    Cost_Exceeded_ICU_Total:                        linctr;
    Cost_Exceeded_PACU_Total:                       linctr;
end-declarations

!Calculating total expected costs of performing the elective cases within the horizon
Cost_Elective_Horizon_Total :=
    sum (ii in Elective, rr in Rooms, nn in Node_Horizon | exists(binary_actual(ii,rr,nn)))
    Probability_Node(nn) * Cost_Elective(ii,rr,nn) * binary_actual(ii,rr,nn);

!Calculating total expected costs of postponing the elective cases out of horizon
Cost_Elective_NextHorizon_Total :=
    sum (ii in Elective, rr in Rooms, nn in Node | exists(binary_actual(ii,rr,nn)) and Time_Node(nn) = (nPeriod+1))
    Probability_Node(nn) * Cost_Elective(ii,rr,nn) * binary_actual(ii,rr,nn);

!Calculating total expected costs of performing recourse decisions of type 1 (postponing elective patients one time period)
Cost_Recourse_1_Total :=
    sum (ii in Elective, rr in Rooms, nn in Node_Horizon | exists(binary_recourse_1(ii,rr,nn)))
    Probability_Node(nn) * Cost_Recourse_1(ii,Time_Node(nn),rr) * binary_recourse_1(ii,rr,nn);

!Calculating total expected costs of performing recourse decisions of type 2 (changing OR for elective patients)
Cost_Recourse_2_Total :=
    sum (ii in Elective, rr in Rooms, nn in Node_Horizon | exists(binary_recourse_1(ii,rr,nn)))
    Probability_Node(nn) * Cost_Recourse_2(ii,Time_Node(nn),rr) * binary_recourse_2(ii,rr,nn);

!Calculating total expected costs of overtime
Cost_Overtime_Total :=
    sum(rr in Rooms, nn in Node_Horizon, pp in PriceRange)
    Probability_Node(nn) * Cost_Overtime(Time_Node(nn),rr,pp) * overtime(rr,nn,pp);

!Calculating total expected costs of exceeding the pre-op capacity
Cost_Exceeded_PREOP_Total :=
    sum(nn in Node_Horizon)
    Probability_Node(nn) * Cost_PREOP(Time_Node(nn)) * exceeded_PREOP(nn);

!Calculating total expected costs of exceeding the ICU capacity
Cost_Exceeded_ICU_Total :=
    sum(nn in Node_Horizon)
    Probability_Node(nn) * Cost_ICU(Time_Node(nn)) * exceeded_ICU(nn);

!Calculating total expected costs of exceeding the PACU capacity
Cost_Exceeded_PACU_Total :=
    sum(nn in Node_Horizon)
    Probability_Node(nn) * Cost_PACU(Time_Node(nn)) * exceeded_PACU(nn);

!Calculating total expected costs (objective function)
Costs :=
    Cost_Elective_Horizon_Total
    + Cost_Elective_NextHorizon_Total
    + Cost_Recourse_1_Total
    + Cost_Recourse_2_Total
    + Cost_Overtime_Total
    + Cost_Exceeded_PREOP_Total
    + Cost_Exceeded_ICU_Total
    + Cost_Exceeded_PACU_Total;

!--------------------------------------------------------------     CONSTRAINTS      ------------------------------------------------------------------------

!Declaring constraints
declarations
    Constraint_Elective_Actual_1:   array(Elective, Rooms, Node)            of linctr;
    Constraint_Elective_Actual_2:   array(Elective, Rooms, Node)            of linctr;
    Constraint_Elective_Actual_3:   array(Elective, Rooms, Node)            of linctr;
    Constraint_Overtime:            array(Rooms, Node_Horizon)              of linctr;
    Constraint_Elective:            array(Elective)                         of linctr;
    Constraint_Emergency:           array(Node_Horizon, Emergency)          of linctr;
    Constraint_Recourse:            array(Elective, Node_Horizon)           of linctr;
```

```
    Constraint_Capacity_PreOp:      array(Node_Horizon)                      of linctr;
    Constraint_Capacity_ICU:        array(Node_Horizon)                      of linctr;
    Constraint_Capacity_PACU:       array(Node_Horizon)                      of linctr;
    Constraint_NonNeg_Overtime:     array(Rooms, Node_Horizon, PriceRange)   of linctr;
    Constraint_UpperBound_Overtime: array(Rooms, Node_Horizon, PriceRange)   of linctr;
    Constraint_NonNeg_E_PreOp:      array(Node_Horizon)                      of linctr;
    Constraint_NonNeg_E_ICU:        array(Node_Horizon)                      of linctr;
    Constraint_NonNeg_E_PACU:       array(Node_Horizon)                      of linctr;
end-declarations

!Constraints (1): Calculating the binary actual variable (1/3), giving information about the elective patients actually treated
forall(ii in Elective, rr in Rooms, nn in Node | exists(binary_actual(ii,rr,nn)) and Time_Node(nn) = Release_Time(ii)) do
    Constraint_Elective_Actual_1(ii,rr,nn) :=
    binary_elective(ii,Time_Node(nn),rr) + binary_recourse_2(ii,rr,nn)
    -(sum(rs in Rooms) binary_recourse_1(ii,rs,nn))
    -(sum(rt in Rooms | rt <> rr) binary_recourse_2(ii,rt,nn))
    <= binary_actual(ii,rr,nn);
end-do

!Constraints (2): Calculating the binary actual variable (2/3), giving information about the elective patients actually treated
forall(ii in Elective, rr in Rooms, nn in Node_Horizon | exists(binary_actual(ii,rr,nn)) and Time_Node(nn) > Release_Time(ii)) do
    Constraint_Elective_Actual_2(ii,rr,nn) :=
    binary_elective(ii,Time_Node(nn),rr) + binary_recourse_1(ii,rr,Preceding_Node(nn)) + binary_recourse_2(ii,rr,nn)
    -(sum(rs in Rooms) binary_recourse_1(ii,rs,nn))
    -(sum(rt in Rooms | rt <> rr) binary_recourse_2(ii,rt,nn))
    <= binary_actual(ii,rr,nn);
end-do

!Constraints (3): Calculating the binary actual variable (3/3), giving information about the elective patients actually treated
forall(ii in Elective, rr in Rooms, nn in Node | exists(binary_actual(ii,rr,nn)) and (nPeriod+1) = Time_Node(nn)) do
    Constraint_Elective_Actual_3(ii,rr,nn) :=
    binary_elective(ii,Time_Node(nn),rr) + binary_recourse_1(ii,rr,Preceding_Node(nn))
    <= binary_actual(ii,rr,nn);
end-do


!Constraints (4): Calculating the overtime variables, measuring by how much the regular capacity is exceeded
forall (rr in Rooms, nn in Node_Horizon) do
    Constraint_Overtime(rr,nn) :=
    sum(ii in Elective) (Time_Elective(ii,nn) * binary_actual(ii,rr,nn))
    + sum(jj in Emergency_t_set(nn)) (Time_Emergency(jj,nn) * binary_emergency(jj,rr,nn))
    <= (Regular_Capacity(rr,nn) + sum(pp in PriceRange) overtime(rr,nn,pp));
end-do

!Constraints (5): Making sure that each elective case is assigned to one time period and one room
forall (ii in Elective) do
    Constraint_Elective(ii) :=
    sum(rr in Rooms) sum(tt in Horizon_C | exists(binary_elective(ii,tt,rr)))
    binary_elective(ii,tt,rr)
    = 1;
end-do

!Constraints (6): Making sure that each emergency case is assigned to a room in the time period of arrival
forall (nn in Node_Horizon, jj in Emergency_t_set(nn)) do
    Constraint_Emergency(nn,jj) :=
    sum(rr in Rooms | exists(binary_emergency(jj,rr,nn)))
    (binary_emergency(jj,rr,nn))
    = 1;
end-do

!Constraints (7): Controlling the recourse variables:
forall (ii in Elective, nn in Node_Horizon) do
    Constraint_Recourse(ii,nn) :=
    sum (rr in Rooms | exists(binary_elective(ii,Time_Node(nn),rr)))
    (binary_elective(ii,Time_Node(nn),rr) - binary_recourse_1(ii,rr,nn) - binary_recourse_2(ii,rr,nn))
    >= 0;
end-do
```

```
!Constraints (8): Calculating exceeded capacity of pre-op variable
forall (nn in Node_Horizon) do
    Constraint_Capacity_PreOp(nn) :=
    sum(ii in Elective, rr in Rooms_Elective_set(ii) | exists(binary_elective(ii,Time_Node(nn),rr)) and Time_Node(nn) = Release_Time(ii))
        binary_elective(ii,Time_Node(nn),rr)
    + sum(ii in Elective, rr in Rooms_Elective_set(ii) | exists(binary_elective(ii,Time_Node(nn),rr)) and Time_Node(nn) > Release_Time(ii))
     (binary_elective(ii,Time_Node(nn),rr) + binary_recourse_1(ii,rr,Preceding_Node(nn)))
    + Emergency_t_amount(nn)
    <= (Capacity_PREOP(Time_Node(nn)) + exceeded_PREOP(nn));
end-do

!Constraints (9): Calculating exceeded capacity of ICU variable
forall (nn in Node_Horizon) do
    Constraint_Capacity_ICU(nn) :=
    (sum (ii in Elective_ICU_set)
    sum(rr in Rooms)
    sum (nm in Earlier_Nodes_set(nn) | (Duration_PO_Elective(ii,nm) + Time_Node(nm)) >= Time_Node(nn))
    binary_actual(ii,rr,nm)) + Emergency_t_ICU(nn)
    <= (Capacity_ICU(Time_Node(nn)) + exceeded_ICU(nn));
end-do

!Constraint (10): Calculating exceeded capacity of PACU variable
forall (nn in Node_Horizon) do
    Constraint_Capacity_PACU(nn) :=
    (sum (ii in Elective_PACU_set)
    sum(rr in Rooms)
    sum (nm in Earlier_Nodes_set(nn) | (Duration_PO_Elective(ii,nm) + Time_Node(nm)) >= Time_Node(nn))
    binary_actual(ii,rr,nm)) + Emergency_t_PACU(nn)
    <= (Capacity_PACU(Time_Node(nn)) + exceeded_PACU(nn));
end-do

!Constraints (11)-(15) are already included (done when the variables were created).

!Constraints (16): Defining the bounds of the overtime variables
forall(nn in Node_Horizon, rr in Rooms, pp in PriceRange | exists(overtime(rr,nn,pp))) do
    Constraint_NonNeg_Overtime(rr,nn,pp) :=
    overtime(rr,nn,pp) >= 0;
    Constraint_UpperBound_Overtime(rr,nn,pp) :=
    overtime(rr,nn,pp) <= PriceRange_Capacity(rr,nn,pp);
end-do

!Constraints (22)-(24): Defining non-negativity of the exceeded_PREOP, exceeded_ICU, and exceeded_PACU variables
forall(nn in Node_Horizon) do
    Constraint_NonNeg_E_PreOp(nn) :=
    exceeded_PREOP(nn) >= 0;
    Constraint_NonNeg_E_ICU(nn) :=
    exceeded_ICU(nn) >= 0;
    Constraint_NonNeg_E_PACU(nn) :=
    exceeded_PACU(nn) >= 0;
end-do

!-------------------------------------------------------------    OUTPUT - TEMPORARY VALUES  ---------------------------------------------------------------

fopen(Output, F_OUTPUT+F_APPEND);
writeln('Datafile:      ', DataFile);
writeln('Comments:      ', Comments);
writeln;
writeln('Output is written to:      ',Output);
writeln('The corresponding variable values of the best solution are written to:     ',DataFileFix);
writeln;
writeln('SOLUTION PROCESS:');
writeln("Time:  Best Bound: Solution value:");
fclose(F_OUTPUT);

procedure printsol
 declarations
```

```
  tempobjval:real;
  temptime:real;
  tempbestbound:real;
 end-declarations

 tempobjval:= getparam("XPRS_lpobjval");
 temptime:= gettime;
 tempbestbound:= getparam("XPRS_bestbound");
 fopen(Output, F_OUTPUT+F_APPEND);
 writeln(temptime," ",tempbestbound,"   ",tempobjval);
 fclose(F_OUTPUT);
end-procedure



!----------------------------------------------------------------     OPTIMIZATION    ---------------------------------------------------------------------------------

setparam("XPRS_MAXTIME",Maxtime);           ! The maximum time in seconds that the optimizer will run before it terminates
setparam("XPRS_MIPRELSTOP",0.005);          ! Gap tolerance (relative distance from theoretical optimum)
!setparam("XPRS_NODESELECTION",5);          ! Modify Optimizer control parameter NODESELECTION - 5 = pure depth first
!setparam("XPRS_VERBOSE", true);            ! To print temporary output

!Results from tuning
setparam("XPRS_DEFAULTALG",3);              ! PRIMAL SIMPLEX
setparam("XPRS_CUTFREQ",2);
setparam("XPRS_HEURDIVESTRATEGY",5);
setparam("XPRS_HISTORYCOSTS",0);
setparam("XPRS_MIPPRESOLVE",3);
setparam("XPRS_PERTURB",0.0001);
setparam("XPRS_ROOTPRESOLVE",1);
setparam("XPRS_SBEFFORT",0.25);
setparam("XPRS_SBESTIMATE",5);

setcallback(XPRS_CB_INTSOL, "printsol");    ! Print best solutions during run

minimize(Costs);

!----------------------------------------------------------------     OUTPUT     ---------------------------------------------------------------------------------

declarations
    Expected_Overtime:              array(Rooms, PriceRange, Horizon)   of linctr;
    Expected_Exceeded_PREOP:        array(Horizon)                      of linctr;
    Expected_Exceeded_ICU:          array(Horizon)                      of linctr;
    Expected_Exceeded_PACU:         array(Horizon)                      of linctr;
    stoptime:                                                               real;
    runningtime:                                                            real;
end-declarations

stoptime := gettime;
runningtime:=(stoptime-starttime);      !Calculate running time

!Calculating expected overtime hours
forall (rr in Rooms, pp in PriceRange, tt in Horizon) do
    Expected_Overtime(rr,pp,tt) :=
    sum (nn in Node_Horizon | Time_Node(nn)=tt)
    Probability_Node(nn) * overtime(rr,nn,pp);
end-do

!Calculating expected amount of patients exceeding pre-op capacity
forall (tt in Horizon) do
    Expected_Exceeded_PREOP(tt) :=
    (sum (nn in Node_Horizon | Time_Node(nn)=tt)
    (Probability_Node(nn) * exceeded_PREOP(nn)) );
end-do

!Calculating expected amount of patients exceeding ICU capacity
forall (tt in Horizon) do
```

```
    Expected_Exceeded_ICU(tt) :=
    (sum (nn in Node_Horizon | Time_Node(nn)=tt)
    (Probability_Node(nn) * exceeded_ICU(nn)) );
end-do

!Calculating expected amount of patients exceeding PACU capacity
forall (tt in Horizon) do
    Expected_Exceeded_PACU(tt) :=
    (sum (nn in Node_Horizon | Time_Node(nn)=tt)
    (Probability_Node(nn) * exceeded_PACU(nn)) );
end-do

!-----------------------

!Output to Output file
!fopen("tee:xpress/ORSP_output/120504_output.txt&", F_OUTPUT+F_APPEND);          ! Activate if you want output to be printed in window + txt file
fopen(Output, F_OUTPUT+F_APPEND);
writeln;
writeln;
!writeln('Fixed variables?      ', Fixed_binary_elective);
writeln('Total running time:        ', getsol(runningtime));
writeln;
writeln('The average expected costs are minimized to        ', getobjval);
writeln('Best bound:        ', getparam("XPRS_bestbound"));
writeln;
writeln('Expected costs of..');
writeln('   ..performing the elective cases:     ', getsol(Cost_Elective_Horizon_Total));
writeln('   ..postponing elective cases to the next planning horizon:   ', getsol(Cost_Elective_NextHorizon_Total));
writeln('   ..postponing the elective cases one time period (recourse 1):   ', getsol(Cost_Recourse_1_Total));
writeln('   ..changing the operating rooms of the elective cases (recourse 2):  ', getsol(Cost_Recourse_2_Total));
writeln('   ..total overtime:   ', getsol(Cost_Overtime_Total));
writeln('   ..exceeding the pre-op capacity:    ', getsol(Cost_Exceeded_PREOP_Total));
writeln('   ..exceeding the ICU capacity:   ', getsol(Cost_Exceeded_ICU_Total));
writeln('   ..exceeding the PACU capacity:  ', getsol(Cost_Exceeded_PACU_Total));
writeln;
writeln;

forall (tt in Horizon) do
    writeln('TIME PERIOD:   ', tt);
    writeln;
    writeln('   Elective cases scheduled to be done:');
    writeln('   ------------------------------------');
    writeln;
    writeln('   room #   |');
    forall(rr in Rooms) do
        write('     ', rr, '|   ');
            forall(ii in Elective) do
                if getsol(binary_elective(ii,tt,rr)) = 1 then
                write(ii, ', ');
                end-if
            end-do
        writeln;
    end-do
    writeln;
    writeln('   Expected overtime hours per price range:    ');
    write('             ');
    forall (pp in PriceRange) do
        write(pp, '  ');
    end-do
    writeln;
    forall (rr in Rooms) do
        write('     room ', rr, ':    ');
        forall (pp in PriceRange) do
            write(strfmt(getsol(Expected_Overtime(rr,pp,tt)),5,2), '    ');
        end-do
        writeln;
    end-do
```

```
    writeln;
    writeln('   Expected amount of passengers exceeding the capacity of');
    writeln('       pre-op:   ', getsol(Expected_Exceeded_PREOP(tt)));
    writeln('       ICU:      ', getsol(Expected_Exceeded_ICU(tt)));
    writeln('       PACU :    ', getsol(Expected_Exceeded_PACU(tt)));
    writeln;
end-do

writeln;
write('Elective cases scheduled to be done in the next planning horizon:    ');
writeln;
write('     ');
forall(rr in Rooms) do
    forall(ii in Elective) do
        if getsol(binary_elective(ii,(nPeriod+1),rr)) = 1 then
                write(ii, ', ');
        end-if
    end-do
end-do

writeln;
writeln;
write('******************************************************************************************************');
writeln;
writeln;

writeln('Value_binary_elective: [');
forall (ii in Elective) do
    forall (tt in Horizon_C | tt >= Release_Time(ii)) do
        forall (rr in Rooms_Elective_set(ii)) do
            writeln('   (',ii,',',tt,',',rr,') ',round(getsol(binary_elective(ii,tt,rr))));
        end-do
    end-do
end-do
writeln(']');

writeln;
writeln;
write('--------------------------------------------------------------------------------------------------------------------------');
writeln;
fclose(F_OUTPUT);

!Output to FIX file
fopen(DataFileFix, F_OUTPUT+F_APPEND);
writeln('!Best solution found for datafile ',DataFile);
writeln('Temp_Costs : ',getobjval);
writeln('Fixed_Elective: [');
forall (ii in Elective) do
    forall (tt in Horizon_C | tt >= Release_Time(ii)) do
        forall (rr in Rooms_Elective_set(ii)) do
            writeln('   (',ii,',    ',tt,', ',rr,') ',round(getsol(binary_elective(ii,tt,rr))));
        end-do
    end-do
end-do
writeln(']');
writeln;
writeln;
write('--------------------------------------------------------------------------------------------------------------------------');
writeln;
writeln;
writeln;
fclose(F_OUTPUT);

end-model
```

Bibliography

[1] K. Akartunali, A. J. Miller. A heuristic approach for big bucket multi-level production planning problems. *European Journal of Operational Research*, 193(2):396-411, 2009.

[2] D. Barro, E. Canestrelli. Time and nodal decomposition with implicit non-anticipativity constraints in dynamic portfolio optimization. *Mathematical Methods in Economics and Finance*, 1:1-20, 2006.

[3] T. Bhattacharyya, M.S. Vrahas, S.M. Morrison, E. Kim, R.A. Wiklund, R.M. Smith, and H.E. Rubash. The value of the dedicated orthopaedic trauma operating room. *The Journal of TRAUMA Injury Infection and Critical Care*, 60(6):1336-1341, 2006.

[4] J. Birge and F. Louveaux. Introduction to stochastic programming. *Springer Verlag*, 2007.

[5] J. Bowers and G. Mould. Managing uncertainty in orthopaedic trauma theatres. *European Journal of Operational Research*, 154:599-608, 2004.

[6] B. Cardoen, E. Demeulemeester, J. Beliën. Operating room planning and scheduling: A literature review. *European Journal of Operational Research*, 201(3):921-932, 2010.

[7] E. Danna, E. Rothberg, C. Le Pape. Exploring relaxation induced neighborhoods to improve MIP solutions. *Mathematical Programming A*, 102(1):71-90, 2005.

[8] C. Dillenberger, L. F. Escudero, A. Wollensak, W. Zhang. On practical resource allocation for production planning and scheduling with period overlapping setups. *European Journal of Operational Research*, 75:275-286, 2009.

[9] L. Escudero, A. Garín, M. Merino, and G. Pérez. The value of the stochastic solution in multistage problems. *Top*, 15(1):48-64, 2007.

[10] M. Fischetti, A. Lodi. Local branching. *Mathematical Programming B*, 98(1):23-47, 2003.

[11] R. Gabel, J. Kulli, B. S. Lee, D. G. Spratt, D. S. Ward. Operating Room Management. *Butterworth-Heinemann*, 1999.

[12] D. Gupta. Surgical suites' operations management. *Production and Operations Management*, 16(6):689-700, 2007.

[13] J.L. Higle. Stochastic Programming: Optimization When Uncertainty Matters. *INFORMS Tutorial paper*, 2005.

[14] K.-H. Krempels and A. Panchenko. An approach for automated surgery scheduling. *Proceedings of the Sixth International Conference on the Practice and Theory of Automated Timetabling*, 2006.

[15] M. Lamiri, X. Xie, A. Dolgui, and F. Grimaud. A stochastic model for operating room planning with elective and emergency demand for surgery. *European Journal of Operational Research*, 185:1026-1037, 2008.

[16] D.-N. Pham and A. Klinkert. Surgical case scheduling as a generalized job shop scheduling problem. *European Journal of Operational Research*, 185:1011-1025, 2008.

[17] Y. Pochet, L. A. Wolsey. Production Planning by Mixed Integer Programming. *Springer Series in Operations Research and Financial Engineering* 2006.

[18] F. Schroyen. Norsk helsesektor: En sektor i vekst - Norge Handelshøyskole *Silhuetten*, 1, 2007.

[19] J. Sun, X. Liu. Scenario formulation of stochastic linear programs and the homogenous self-dual interior point method. *INFORMS Journal on Computing*, 18(4):444-454, 2006.

[20] K. T. Uggen, M. Fodstad, V. S. Nørstebø. Using and extending fix-and-relax to solve maritime inventory routing problems. *Top*, 1-23, 2011.

[21] G. Wullink, M. Van Houdenhoven, E.W. Hans, J.M. van Oostrum, M. van der Lans, and G. Kazemier. Closing Emergency Operating Rooms Improves Efficiency. *Journal of Medical Systems*, 31:543-546, 2007.

[22] Phone call with Atle Riise in Sintef, May 4, 2012.

[23] OECD Health Data 2011 - A selection of key indicators. Available from http://www.oecd.org

[24] Helseregnskap, 1997-2011. Available from http://www.ssb.no/helsesat