Simulation-Based Rolling Horizon Scheduling for Operating Theatres

Andersen, Anders Reenberg; Stidsen, Thomas Jacob Riis; Reinhardt, Line

Published in:
SN Operations Research Forum

DOI:
10.1007/s43069-020-0009-6

Publication date:
2020

Document Version
Peer reviewed version

Citation for published version (APA):
Andersen, A. R., Stidsen, T. J. R., & Reinhardt, L. (2020). Simulation-Based Rolling Horizon Scheduling for Operating Theatres. *SN Operations Research Forum*, *1*(2), [9]. https://doi.org/10.1007/s43069-020-0009-6

Noname manuscript No.
(will be inserted by the editor)

Simulation-based Rolling Horizon Scheduling for Operating Theatres

Anders Reenberg Andersen · Thomas Jacob Riis Stidsen · Line Blander Reinhardt

Received: January 10th, 2020 / Accepted: March 6th, 2020

Abstract Daily scheduling of surgical operations is a complicated and recurrent problem in the literature on health care optimization. In this study, we present an often overlooked approach to this problem that incorporates a rolling and overlapping planning horizon. The basis of our modeling approach is a Markov decision process, where patients are scheduled to a date and room on a daily basis. Acknowledging that both state and action space are only partially observable, we employ our model using a simulation-based method, where actions are derived from a heuristic search procedure.

We test the potential of using this modeling approach on the resulting hospital costs, and number of patients that are outsourced to avoid violating constraints on capacity. Using data from a Danish hospital, we find a distinct improvement in performance when compared to a policy that resembles a manual planner. Further analysis shows that substantial improvements can be attained by employing other simple policies.

F. Author (Corresponding Author)
Department of Applied Mathematics and Computer Science, Technical University of Denmark
Anker Engelunds Vej 1, 2800 Kgs. Lyngby, Denmark
Tel.: +45 45254800
E-mail: arean@dtu.dk

S. Author
Department of Engineering Management, Technical University of Denmark
Anker Engelunds Vej 1, 2800 Kgs. Lyngby, Denmark
Tel.: +45 45254800
E-mail: thst@dtu.dk

T. Author
Department of People and Technology, Roskilde University
Universitetsvej 1, 4000 Roskilde, Denmark
Tel.: +45 46743497
E-mail: liner@ruc.dk

Keywords Patient scheduling · Stochastic optimization · Decision processes · Heuristics

1 Introduction

Surgical procedures are one of the key elements of a hospital and involves many different clinical specializations from organ to orthopedic surgery.

According to statistics from the Organisation for economic co-operation and development [14], the number of surgical procedures is increasing relative to the population size across several countries. Australia, Canada, Finland, and the United Kingdom have experienced an increase of roughly 50% over the past 20 years. For Austria and Denmark, the increase is roughly 100%, and in Portugal, the increase is roughly 350% over the same period.

It seems reasonable that the use of resources in this part of the hospital has received a substantial amount of attention in the Danish health care sector. In September 2015, the Danish Ministry of Health [22] published a report on the overall status of the public health care sector, showing that waiting time for surgery has been increasing for a fourth of the investigated departments in the period from 2011 to 2014. Other governmental reports suggest a lack of resource utilization for Operating Theatres (OTs) as well. In March 2015, the National Audit Office of Denmark [21] published a report on the use of staff resources based on four departments in orthopedic surgery with the conclusion that staff working hours are not ensured to be fully utilized for a majority of cases.

On the other hand, ensuring an efficient utilization of surgical resources can be quite intractable. To attain an efficient use of both staff and equipment resources, several decisions on multiple organizational levels have to be considered [15, 18]. These range from long-horizon planning problems, such as deciding on the overall required capacity, to day-to-day scheduling (and re-scheduling) of patients.

From interviews, we found that manual planners have a time-consuming and complicated task at hand. In our case hospital, planners have to ensure that equipment in the room is compatible with the surgical procedure. Simultaneously, planners must ensure that patients are treated by the same surgeon they were examined by *and* that the waiting time does not violate a hard upper limit. Therefore, including overtime-costs and capacity efficiency yields an impractical if not impossible task for the manual planner.

Our objective in this study is to provide hospital planners with a decision tool capable of optimizing the scheduling of patients for operation, respecting the constraints that are relevant to the hospital. In our case, we consider that patients are scheduled on a day-to-day basis and require that a rolling and overlapping planning horizon is taken into account. Thus, the decisions that are made on each day have to be anticipative.

Our methodological approach will be a simulation-based Markov Decision Process (MDP) that minimizes the long-term costs of overtime and setting up operating rooms.

In Section 2 we describe the scheduling problem in details. Next, in Section 3 we present our simulation-based MDP and describe how an allocation of patients can be derived using this approach. In Section 4 we apply our approach to data from a Danish hospital and assess our MDP implementation by comparing to other scheduling methods. Finally, we present our conclusion and suggestions for future work in Section 5.

1.1 Literature Review

The problem of OT planning is a recurrent topic that has been covered by a substantial amount of papers. There exists several surveys on the subject of which some of the recent have been conducted by Cardoen et al., 2010 [5], Guerriero & Guido, 2011 [15], May et al., 2011 [18], and lately Samudra et al., 2016 [25] where 137 journal papers on the subject of OT planning were found in the period of 2004 to 2014.

With respect to the organizational decision levels, Guerriero & Guido, 2011 [15] find that the studies can be classified into three categories: Strategic (long-term decisions), tactical (medium-term decisions), and operational (short-term decisions). May et al., 2011 [18] add further three decision levels denoted: Very long-term, very short-term, and contemporaneous. The decisions relevant to the very long-term are related to the layout of physical resources [32], such as the construction of operating rooms. Long-term decisions are related to patient flow patterns and assigning overall capacity to surgical groups [3, 29]. Medium-term decisions involve defining the master surgical schedule, where the clinical specializations are assigned to specific rooms and time-windows [31]. On the short-term, the patient procedures are assigned to a specific time and room on a day-to-day basis, and on the very short-term and contemporaneous level, last-minute scheduling and re-scheduling is conducted [1, 4, 9–12, 17, 20, 23, 27].

Focusing on the short-term operational level of OT planning, we have found a range of different approaches and problem structures. Studies can mainly be categorized into considering completely deterministic "off-line" problems [4, 6, 12,30,33], to incorporating uncertainty features such as random procedure time [1, 9, 17] and disruptions caused by emergency demand [11, 17]. Surprisingly, we only encountered two studies on the allocation of patients where stochastic future arrivals were accounted for [23, 34]. However, Samudra et al., 2016 [25] shows that incorporating stochasticity constitutes more than half of the papers on OT planning.

The specific modeling approaches of short-term OT planning range from mathematical programming and heuristics [1, 4, 9, 11, 12, 30, 33] to Discrete-event and Monte Carlo simulation [10, 27], and further to a mixture of these

[17]. For the purely deterministic cases Xiang et al., 2015 [33] and Van Huele & Vanhoucke, 2014 [30] combine the surgical scheduling problem with a staff rostering problem. Xiang et al., 2015 [33] develop a modified Ant Colony Optimization algorithm and test the model by using both data from the literature and real data from a Chinese hospital. Van Huele & Vanhoucke, 2014 [30] approach the problem by using Mixed Integer Linear Programming (MILP) based on the most frequent objectives and constraints from the literature. In Fei et al., 2010 [12] and Cardoen et al., 2009 [4] the focus is more on the scheduling and sequencing of the surgical procedures. Fei et al., 2010 [12] use an approach comprising two phases where patients are firstly assigned a date by using a column-generation-based heuristic, and subsequently sequenced by using a hybrid genetic algorithm. Cardoen et al., 2009 [4] focus on the sequencing of procedures and develop MILP models which lead to either exact or heuristic solutions.

For the studies that incorporate uncertainty, Batun et al., 2011 [1] and Lamiri et al., 2008 [17] use Stochastic Programming (SP) to minimize the total cost of scheduling patients over a planning horizon. Specifically, Batun et al., 2011 [1] develops a two-stage stochastic MILP and investigates the impact of parallel surgery processing and pooling operating rooms. Related hereto, Lamiri et al., 2008 [17] develops an SP model, and moreover a method combining Monte Carlo simulation and a MIP model to schedule elective patients within a specific planning horizon, and emergency patients on the same day of arrival.

Methods based on MILP modeling can in some cases become too inefficient as found by Erdem et al., 2012 [11], where a MILP model and Genetic Algorithm (GA) are developed to reschedule elective patients upon the arrival of emergency patients. For the MILP model, Erdem et al., 2012 [11] finds that a commercial solver is sufficient for only a limited "light" case, and therefore develops a GA to find solutions close to optimality for the more complex cases. In addition, Denton et al., 2007 [9] focus on heuristic methods for deriving the sequencing of patients in operating rooms, and find that a simple sequencing rule can be used to optimize both waiting time and overtime-costs.

As the above shows, random duration of procedures and the impact from emergency arrivals draws a lot of attention, but there is limited focus on overlapping planning horizons originating from uncertain future arrivals. However, Range et al., 2016 [23] schedule elective patients based on a MILP model and solve the problem with column generation. Future arrivals are accounted for by measuring the expected number of future patients who cannot be scheduled for surgery.

Another study that accounts for uncertain future arrivals is Zhang et al., 2019 [34]. Similar to the approach by Fei et al., they use a model that consists of two phases. In the first phase, future decisions are incorporated by selecting unscheduled patients with an MDP that minimizes the expected long-term costs. In the second phase, the selected patients are finally assigned to the respective surgical blocks with an SP model.

In this study, we also consider the problem of allocating patients to a day and room as a sequential decision problem with overlapping planning horizons. We assume that surgical operations can begin at any time within the opening hours of the operating room and even stretch into overtime. Our model is a heuristic approach that accounts for random inter-arrival times, and procedure duration, and is based on a simulation-based MDP that derives an allocation of patients in one phase by minimizing the combined long-term costs of overtime and setting up operating rooms.

2 Problem Description

Finding a good schedule yields multiple objectives for the hospital planner. Long waiting times and overcrowding of wards may cause a decrease in both subjective and objective care quality as was found by Hansagi et al., 1992 [16] and McMillan et al., 1986 [19]. To make things even more difficult, planners in our case hospital are required to keep the patient waiting time within a hard upper limit, where insufficient operating room capacity leads to expensive overtime-costs. Outsourcing patients to other units is one way of avoiding these problems but comes with qualitative and logistical costs.

In this study, we consider a hospital where a planner schedules surgical operations on a daily basis. The hospital treats both elective and emergency patients in a range of different clinical specializations, but operating rooms are reserved for each patient type and for each respective area of specialization.

In contrast to other studies, such as Erdem et al., 2012 [11], we assume that emergency resources are rarely insufficient so that we may limit our scope to the scheduling of elective patients only. We further focus on a single clinical specialization and assume that resources are negligibly shared with other specializations.

Patients have continuous random inter-arrival time, but we assume that the hospital planner can postpone allocating the patients until the end of the day. Procedure requests from elective patients occur only on regular workdays; hence the hospital planner has to make a maximum of five decisions a week.

We assume that all patients have an upper limit on waiting time from the moment the surgical operation is requested. Thus, due to uncertainty in procedure duration and inter-arrival times, capacity may in some instances be insufficient so that patients have to be outsourced to an internal or external treatment unit.

An overview of the scheduling problem is depicted in Figure 1.

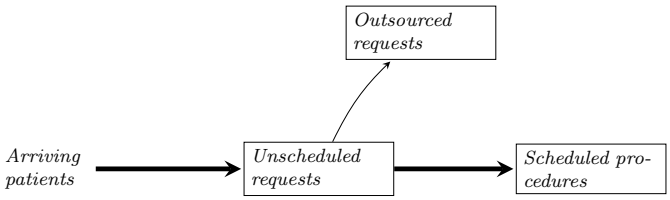

Fig. 1: Overview of the scheduling problem. Requests are unscheduled until the end of the day, after which they will be allocated to a fixed date and room or outsourced to a different treatment unit.

For the remaining of this paper we will refer to scheduled surgical operations as *procedures*. Unscheduled procedures are referred to as *requests*. Days on which the number of requests are positive, such that the hospital planner must decide on an allocation of these, will be referred to as *allocation epochs*.

2.1 Constraints & Dynamics of the Problem

Our aim is to derive a cost-optimized schedule for all requests that have arrived during the day, repeating this process for all future days. When a decision is made, the hospital planner considers a discrete planning horizon of total length, $H \in \mathbb{N}$, such that from the end of the current allocation epoch, $t \in \mathbb{Z}$, all days that are considered in the scheduling problem are $t+1, t+2, \ldots, t+H-1, t+H$.

Let $X \in \mathbb{N}_0$ be a random variable defining the total amount of requests received on an arbitrary day. Then for all days where $X > 0$ a scheduling problem has to be solved with a planning horizon that has been "rolled" accordingly. Let $\delta \in \mathbb{Z}_{>t}$ define the subsequent allocation epoch to t. Further, let Ω_i, where $|\Omega_i| = H$, define the specific set of days contained in the planning horizon of an allocation epoch, i. Hence, if $\delta < t + H$, then $\Omega_t \cap \Omega_\delta \neq \emptyset$, as illustrated in Figure 2.

Let R define a finite set of operating rooms available to the hospital, then the planner has to make a decision involving *both* the finite and discrete planning horizon, and the operating room resources in R. The feasibility in scheduling a procedure for a specific room, $r \in R$, depends on a predefined surgical schedule as well as other constraints which are presented in the following Section 2.1.1. Furthermore, all allocations may induce a cost from setting up the room or when procedures stretch into overtime. Our assumptions related to these costs will be presented in Section 2.1.2.

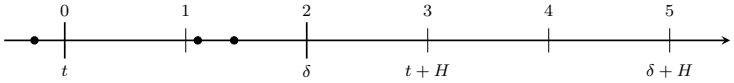

Fig. 2: Example of a rolling and overlapping planning horizon of $H = 3$ days. Requests are illustrated by black dots along the time-line. As a result, planning has to be conducted at $t = 0$ and $\delta = 2$, leading to $\Omega_t = \{1, 2, 3\}$, $\Omega_\delta = \{3, 4, 5\}$ and $\Omega_t \cap \Omega_\delta = \{3\}$.

2.1.1 Constraints

Constraints relevant to the scheduling problem range from the availability of predefined capacity to less tangible factors such as preferences of the staff. In the below, we present each of these constraints separately.

1. **Number of rooms**. The hospital planner can choose to allocate a request to an unopened room provided that this action does not violate an upper limit on open rooms. Let $y_{kl} \in \{0, 1\}$ be 1 if room $k \in R$ is being used on day $l \in T$, where $T = \{t + 1, \cdots, t + H\}$ is the set of workdays within the current planning horizon; and otherwise 0. Further, let $c_l \in \mathbb{N}$ define the maximum number of rooms that is allowed to be opened on day $l \in T$. We assume that the structure of c_l is weekly cyclical such that $c_l = c_{l+5}$. Then, an allocation to a room $k \in R$ on day $l \in T$ is only allowed if subsequently $\sum_{k \in R} y_{kl} \leq c_l$.
2. **Equipment**. Any procedure cannot be allocated to any room, even if c_l is not violated. To account for potential equipment requirements, as well as other preferences that may exist, each procedure type $i \in P$, where P is the set of all procedures that may occur, is constrained to a subset, U_i, of the available rooms, such that $U_i \subseteq R$.
3. **Physicians**. When a request is received by the hospital planner, a specific physician has already been assigned to conduct the procedure. We assume that physician-rosters are not flexible so requests can only be allocated to days for which the physicians are expected to be available at the hospital. Let J define the set of scenarios (or patterns) of days for which physicians will be available. As T is always a finite set, so is J. We assume that a request is randomly assigned to a specific pattern $j \in J$ with known probability.
4. **Opening hours**. Lastly, all operating rooms have a pre-specified time-interval for which they are expected to be open. Let $Y_i \in \mathbb{R}_{>0}$ be a random variable with known distribution that defines the duration of a procedure type, $i \in P$. Furthermore, let $r_{ikl} \in \mathbb{N}_0$ define the number of procedure $i \in P$ that are allocated to room $k \in R$ on day $l \in T$. Then, an allocation to a room $k \in R$ on day $l \in T$ is only allowed if there exists at least one sequence such that all procedures are expected to start within the opening hours. That is, $\sum_{i \in P \setminus \alpha}(r_{ikl} \cdot E[Y_i]) + (\sum_{i \in P}(r_{ikl}) - 1) \cdot m < w_k$, where $m \in \mathbb{R}_{>0}$ is a fixed buffer time, $w_k \in \mathbb{R}_{>0}$ is the time-capacity of

room $k \in R$, and α is the allocated procedure with the longest expected duration for that room and day.

We assume that all allocations are *final* such that each respective procedure is locked in both room and date. Further, as neither of the above constraints are allowed to be violated, and the occurrence of requests is independent from the current schedule, we allow for requests to be outsourced to yield a feasible solution with a maximum number of allocations. In this regard, we assume that allocating all current and future requests is always preferred over outsourcing any of them.

2.1.2 Costs

In combining a suitable schedule, the hospital planner has to consider that there might be a number of implications related to each respective solution. We found from interviews as well as from other studies [27] that some hospitals assess their performance on the utilization of time-capacity for each operating room. Such measure is convenient with respect to day-to-day monitoring and obtaining sufficient data, but does not provide an immediate relation between setting up new rooms and the risk of stretching procedures into overtime. For this reason, we evaluate the implications related to a specific schedule on a sum of "penalties". We refer to these as *costs* as we mainly relate them to direct costs, such as overtime, cleaning, setting up equipment, and so on. We have categorized these costs into two respective groups, presented below:

1. **Setup**. To account for the logistical costs related to equipment and staff preparation we assume that by opening a room the hospital receives a fixed *setup* cost. That is, the setup cost is induced only when the first procedures are allocated to the room, and does otherwise not depend on the utilization of time-capacity.
2. **Overtime**. As mentioned earlier, all procedures are subject to a random duration, and thus are in risk of stretching into overtime. If this is the case, we assume the hospital always pays a supplement to the staff independent of the type of procedure. In addition, some amount of discontent may arise among the staff leading to more errors and a decrease in the treatment quality. As a result we notice that the total penalty related to overtime can be a non-linear increasing function of the duration of overtime.

3 Modeling & Solution Approach

In this section, we present the approach we use to minimize the long-term expected costs of scheduling requests for operation. Our modeling approach is based on a Markov Decision Process (MDP) framework, for which, due to the problem size, we propose a simulation-based "on-line" solution method.

In Section 3.1 we present the specific structure of our modeling approach along with an exact solution method from standard theory. Next, in Section

3.2 we present our solution approach which is based on a simulation-based rollout method resulting in a heuristic policy.

3.1 A Markov Decision Process

Now, recall that we consider a finite set of procedures, $P = \{ProcedureA, ProcedureB, \cdots\}$. Any procedure, $i \in P$, is to be conducted within a fixed planning horizon, $H \in \mathbb{N}$, such that the set of future workdays in the planning horizon are in the set $T = \{t+1, t+2, \cdots, t+H\}$, where $t \in \mathbb{Z}$ is the day from which the planning horizon is observed. Further, let $R = \{Room\ A, Room\ B, \cdots\}$ define the total set of available operating rooms, and $r_{ikl} \in \mathbb{N}_0$ define the number of procedure $i \in P$ that have been scheduled on future day $l \in T$ in room $k \in R$.

In addition, we consider a finite set of all patterns for which physicians can be available within the planning horizon, $J = \{Pattern\ A, Pattern\ B, \cdots\}$. Together with P, these *availability-patterns* make up all of the attributes of any request that may occur. In other words, for any current day let $p_{ij} \in \mathbb{N}_0$ specify the number of requests of type $i \in P$ that are constrained by pattern $j \in J$. Lastly, let $W = \{Monday, Tuesday, \cdots, Friday\}$ define the set of weekdays for which procedures can be allocated, and $d \in W$, then based on the above definitions, we introduce an MDP with state definition,

$$s = [p_{ij}, r_{ikl}, d] \tag{1}$$

divided into three parts: (1) The number and attributes of all current requests, p_{ij}, (2) the amount of each procedure scheduled to future day and room, r_{ikl}, and (3) the current weekday, d. Notice that d can be redundant depending on the structure of the problem from one case to another. If the constraints on room-capacity, c_l, and availability-patterns, J, can be generalized such that they are independent on the type of weekday in $l \in T$, then the state definition can be reduced to $s = [p_{ij}, r_{ikl}]$.

Furthermore, the reader should notice that the value of p_{ij} is generated by a purely stochastic process, whereas the transition into a state with any value of r_{ikl} will always be deterministic in terms of the decision by the planner. Now, let λ_i define the stationary occurrence rate of requests of type $i \in P$, and $X_{ij} \in \mathbb{N}_0$ be a random variable defining the occurrence of a request $i \in P$ constrained by pattern $j \in J$. Then the requests, X_{ij}, are generated according to a multivariate Poisson process with parameters $\lambda_{ij} = \lambda_i \xi_{ij} \quad \forall i, j \in P, J$. Here, $\xi_{ij} \in \mathbb{R}_{0<\xi_{ij}\leq 1}$ is the probability that a request of type $i \in P$ is constrained to pattern $j \in J$; hence $\sum_{j \in J} \xi_{ij} = 1 \quad \forall i \in P$. The assumption that requests are generated by a Poisson process was found adequate by Spratt et al., 2019 [26].

In the following, we present how this modeling approach relates to the action space and transitions of the MDP.

3.1.1 Actions & Transitions

From one day to the next, the MDP changes from a current state $s \in S$ to a new state $s^* \in S$. This *transition* occurs consistently and with fixed time-interval. In addition, for each transition an action has to be chosen from the action space, A_s, available at each *decision epoch* — that is, at the end of every day, where the planner must decide on an allocation of the requests. Let π define a policy such that for any $s \in S$, $\pi(s) = \mathbf{a}$, where $\mathbf{a} \in A_s$. Thus for any arbitrary policy $\pi \in \mathit{\Pi}$, where $\mathit{\Pi}$ is the set of all policies, the MDP will evolve as a Markov chain in discrete time.

Let $\mathbf{a}$ be a vector of the elements $a_{ijkl} \in \mathbb{N}_0$, defining the number of requests of type $i \in P$ constrained by pattern $j \in J$ that are allocated to room $k \in R$ on future day $l \in T$. To account for the outsourcing of requests we further extend $\mathbf{a}$ with the elements $q_{ij} \in \mathbb{N}_0$, defining the number of type $i \in P$ and pattern $j \in J$ that are outsourced. Thus, $\mathbf{a}$ has a total of $|P \times J \times R \times T| + |P \times J|$ elements. The size of A_s is, however, dependent on the values of r_{ikl} in the state s, which is limited by the constraints presented in Section 2.1.1. A_s contains any feasible value of $\mathbf{a}$; hence $1 \leq |A_s| \leq (|R \times T|)^{\sum_{i,j \in P,J} p_{ij}}$.

Notice that $\sum_{k,l \in R,T} a_{ijkl} + q_{ij} = p_{ij} \quad \forall i,j \in P,J$, and as the planning horizon is rolling $r^s_{ikl} + \sum_{j \in J} a_{ijkl} = r^{s^*}_{ik,l-1} \quad \forall i,k,l \in P,R,T \setminus \{t+1\}$, where r^s_{ikl} and $r^{s^*}_{ikl}$ are the schedules for the current state $s \in S$ and subsequent state $s^* \in S$, respectively. Moreover, notice that for $l = t + H$ all rooms are freed such that $r_{ikl} = 0 \quad \forall i,k \in P,R$. However, as procedures are constrained to specific rooms, the only feasible solution may for some cases be to outsource all current requests. If for some decision epoch the number of requests $\sum_{i,j \in P,J} p_{ij} = 0$, then the only action is to let $\sum_{i,j,k,l \in P,J,R,T} a_{ijkl} + q_{ij} = 0$, in which case the MDP merely transitions into the next state resulting in $r^s_{ikl} = r^{s^*}_{ik,l-1} \quad \forall i,k,l \in P,R,T \setminus \{t+1\}$.

Lastly, the transition probability, $\mathbf{p}^{ss^*}_{\mathbf{a}}$, of changing from $s \in S$ to a subsequent $s^* \in S$ by choosing $\mathbf{a} \in A_s$, is merely $\mathbf{p}^{ss^*}_{\mathbf{a}} = Prob\{X_{11} = p_{11}, X_{12} = p_{12}, \cdots, X_{|P||J|} = p_{|P||J|}\}$ if $r^s_{ikl} + \sum_{j \in J} a_{ijkl} = r^{s^*}_{ik,l-1} \quad \forall i,k,l \in P,R,T \setminus \{t+1\}$; otherwise $\mathbf{p}^{ss^*}_{\mathbf{a}} = 0$.

3.1.2 Cost Function

In the previous section we introduced the policy $\pi \in \mathit{\Pi}$, where $\mathit{\Pi}$ is the set of all possible policies for the MDP. Furthermore, recall that for any policy the MDP evolves as a discrete-time Markov chain. Let $V^\pi_\infty(s)$ define the expected long-term costs that are induced by this Markov chain, starting at state $s \in S$, under the policy $\pi \in \mathit{\Pi}$. That is,

$$V^\pi_\infty(s) = E\left[\sum_{t=0}^{\infty} \gamma^t C(s_t, \pi_t(s_t)) \middle| s_0 = s\right] \tag{2}$$

where $C(s_t, \pi_t(s_t))$ is the cost induced from taking action $\pi(s_t)$ in state s_t at time t, $\gamma^t \in \mathbb{R}_{<1}$ a discount factor, and $t = 0$ is any arbitrary point in

time. We define the optimal policy, π^*, as the policy which obtains $V_\infty^{\pi^*}(s) = \min_{\pi \in \Pi} V_\infty^\pi(s) \quad \forall s \in S$, and thus an essential element in minimizing the expected long-term costs is the definition of how each action is penalized through the cost function, $C(s_t, \pi_t(s_t)) = C(s, \mathbf{a})$. The reader should notice that the optimal myopic solution to the scheduling problem is included in the set Π, and thus we have that $V_\infty^{\pi^*}(s) \leq V_\infty^{\pi^\eta}(s)$, where π^η is the policy for which $\pi^\eta(s) = \arg\min_{\mathbf{a} \in A_s} E[\gamma^0 C(s, \mathbf{a})] \quad \forall s \in S$.

Now recall that we consider two different types of costs:

1. A fixed setup cost, $\kappa \in \mathbb{R}_{>0}$, is induced whenever a procedure is scheduled to a new room — that is, whenever $\sum_{i \in P} r_{ikl} = 0$ and $\sum_{i,j \in P,J} a_{ijkl} > 0$ for any $k \in R$ and $l \in T$ in the current state, s.
2. An overtime-cost that accounts for procedures stretching into overtime for any $k \in R$. Let the total capacity utilization of a room be defined by $\tau \in \mathbb{R}_0$, and let $f(\delta)$ define the overtime-cost for an overtime of size $\delta \in \mathbb{R}_0$, where δ is the amount of time that τ exceeds the capacity, w_k, for a room $k \in R$. Now, let $p_k(\tau)$ define the probability density function for a capacity utilization of amount τ in room $k \in R$. We then penalize an action according to the total *expected* amount of overtime, $\sum_{k \in R} o_k$, for the subsequent day, $l = t + 1$, where o_k is defined in (3). Notice that this formulation generalizes to any continuous distribution, $p_k(\tau)$, for which $\tau \geq 0$ and overtime-cost function, $f(\delta)$, for which $\delta \geq 0$.

$$o_k = \int_{w_k}^{\infty} p_k(x) f(x) \cdot dx \tag{3}$$

To ensure that actions are penalized for outsourcing requests, we further introduce a large penalty, $\phi \in \mathbb{R}_{>0}$ for every outsourced request. Finally, the resulting cost function is presented in (4), where y_{kl}^s and $y_{kl}^{s^*}$ is 1 if a room $k \in R$ is scheduled for use on day $l \in T$ in the current state $s \in S$ or subsequent state $s^* \in S$, respectively; and otherwise 0.

$$C(s, \mathbf{a}) = \sum_{k \in R} o_k + \sum_{k,l \in R, T \setminus \{t+H\}} (y_{kl}^{s^*} - y_{k,l+1}^s) \cdot \kappa + \sum_{i,j \in P,J} q_{ij} \cdot \phi \tag{4}$$

3.2 A Heuristic Approach

In our case the size of a single state and especially the state space, S, can be *very* large, even for small problem instances. Assuming a rather limited case where physicians are always available such that $|J| = 1$, procedures are constrained to only one room, and further that $c_l = c_{l+1} \quad \forall l \in T$, leading to $s = [p_i, r_{il}]$, there are a total of $|P| + |P \times T|$ elements in each state. That is, for a case with merely $|P| = 10$ different procedures, and a planning horizon of $|T| = 20$ days, a single state is comprised of 210 elements. Additionally, by

assuming a maximum number, n, of requests per type, $i \in P$, and a capacity limit, m, of procedures per day, the state space would have a total size of $|S| = (n+1)^{|P|} \cdot \left(\frac{1}{|P|!}\prod_{i=1}^{|P|}(m+i)\right)^{|T|-1}$ states — for which a direct implementation of an exact algorithm would be a computational challenge. Furthermore, in the worst case the action space attains a size of $|A_s| = (|R \times T|)^{\sum_{i,j \in P,J} p_{ij}}$. For these reasons, we assume that an analytical approach to solve the MDP would be completely intractable.

The method presented in this section is based on a simulation-based approach. That is, instead of deriving an optimal action $\pi^*(s)$ for each of the states $s \in S$, we only rely on deriving a *good* action for the current state. Our approach is based on a rollout algorithm proposed by Bertsekas & Castañon, 1999 [2], and later extended to parallel rollout by Chang et al., 2004 [8].

Consider some arbitrary allocation epoch, t, in which the requests p_{ij} are scheduled. These requests will be constrained by the occupation of the procedures that are already in the schedule, r_{ikl}, and for any policy induce the long-term cost $V^{\pi}_{\infty}(s)$. Now consider an optimal policy, $\pi \in \Pi$, that has been derived for a finite model-horizon H'. Then as $H' \to \infty$, the policy $\pi \to \pi^*$ for the infinite case. The cost of such a policy would then be,

$$V^{\pi}_{H'}(s) = E\left[\sum_{t=0}^{H'} \gamma^t C(s_t, \pi_t(s_t)) \middle| s_0 = s\right] \tag{5}$$

quite similar to (2). We assume for the remaining of this paper that $\gamma^t = 1$ for $t = 0, 1, \ldots, H'$. From the definition in (5), we note that the decision a hospital planner has to take from the current state s, should be derived from the sum of first the current *known* cost $C(s, \mathbf{a})$, and second an expected long-term cost from a sequence of future actions. Thus, we let a *rollout* policy, π', be defined as the result of a sequence of actions that has been derived under (6),

$$\pi'(s) \in \arg\min_{\mathbf{a} \in A_s} \{C(s, \mathbf{a}) + E[\tilde{V}^{\pi}_{H'-1}(f(s, \mathbf{a}, \omega))]\} \tag{6}$$

where $E[\tilde{V}^{\pi}_{H'-1}(\cdot)]$ approximates (5), and $\tilde{V}^{\pi}_{H'-1}(\cdot)$ represents the total cost of a path of decisions over the horizon $t = 1$ to H'. Further, we let the subsequent state relative to s be defined as $s^* = f(s, \mathbf{a}, \omega)$. That is, the combined result of the current state, s, the action, $\mathbf{a}$, and a random disturbance of the system ω.

Similar to Bertsekas & Castañon, 1999 [2], we fix the disturbances, ω, to a finite set of values such that we limit our scope to a mere sample of the potential subsequent states. That is, we randomly sample N disturbances and then evaluate $\tilde{V}^{\pi}_{H'-1}(f(s, \mathbf{a}, w^j))$ for $j = 1, 2, \ldots, N$, yielding the N paths illustrated in Figure 3. Thus, for the decision of choosing an action in the rollout policy, π', (6) changes to (7).

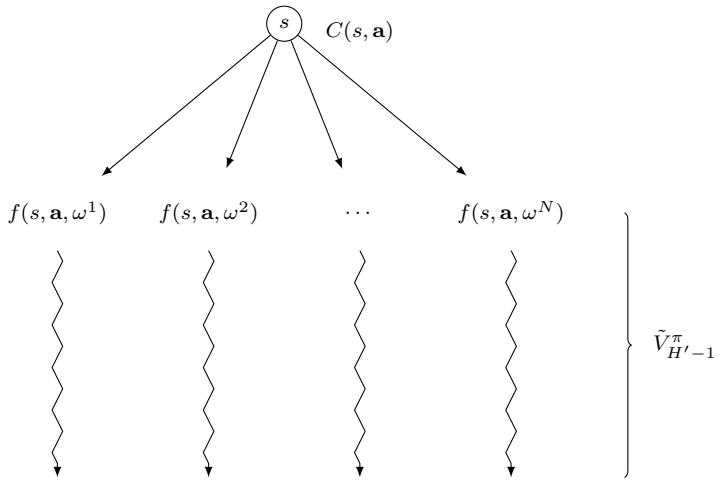

Fig. 3: For the expression in (7), these are the subsequent paths and costs that are observed from the current state s.

$$\pi'(s) \in \arg\min_{\mathbf{a} \in A_s} \{C(s, \mathbf{a}) + \frac{1}{N} \sum_{j=1}^{N} \tilde{V}^{\pi}_{H'-1}(f(s, \mathbf{a}, \omega^j))\} \tag{7}$$

Notice that in practice, w^j is sampled by using pseudo-random numbers that are then converted into obtaining the requests, p_{ij}, at each subsequent state.

How to evaluate $\tilde{V}^{\pi}_{H'-1}(f(s, \mathbf{a}, \omega))$ will be presented in the following Section 3.2.1. Moreover, the expression in (7) requires a full enumeration of the state dependent action space A_s. As mentioned previously, the size of A_s can be quite intractable, and therefore we require a robust search procedure to reduce the computational requirements. We present this procedure in Section 3.2.2.

3.2.1 Simulation-based Value Evaluation

Let Λ define a non-empty finite set of policies that all perform well for the hospital scheduling problem. By choosing the one policy that performs the best related to the current state, s, we allow for a rollout policy that continually adapts to the system. This is the basis of parallel rollout [8]. A related approach is to choose an action based on the current *weighted* average performance of the policies in Λ, which is the method we will employ in this study. We base our approach on a Simulated Annealing Multiplicative Weights (SAMW) algorithm proposed by Chang et al., 2007 [7]. Let $\phi(\pi)$ define the weighting of policy $\pi \in \Lambda$, such that $\sum_{\pi \in \Lambda} \phi(\pi) = 1$. The aim of the SAMW algorithm is then to concentrate the weighting on the *currently* (related to s) best policies

in Λ.

Let $\phi^i(\pi)$ define the weight of policy π at iteration i. Then,

$$\phi^{i+1}(\pi) = \phi^i(\pi)\frac{\beta_i^{-\tilde{V}_i^\pi}}{Z^i} \tag{8}$$

where $\tilde{V}_i^\pi$ corresponds to $\tilde{V}_{H'-1}^\pi(f(s,\mathbf{a},\omega^j))$ at iteration i for any of the disturbances ω^j. In addition, we have that $\pi \in \Lambda$ and $\beta_i \in \mathbb{R}_{>1}$ is a "cooling" parameter that decreases as function of iteration i. Furthermore, Z^i is a normalization parameter,

$$Z^i = \sum_{\pi\in\Lambda} \phi^i(\pi)\beta_i^{-\tilde{V}_i^\pi} \tag{9}$$

Now, we let $\omega_1^j, \omega_2^j, \ldots, \omega_{H'}^j$, where $\omega^j = \omega_1^j$, define a *path* of random disturbances such that we get $\tilde{V}_i^\pi = \sum_{t=1}^{H'} C(s_t, \pi(s_t))$, where $s_t = f(s_{t-1}, \pi(s_{t-1}), \omega_t^j)$, s_0 is the current state s, and $\pi(s_0)$ is the current action $\mathbf{a}$. In each iteration we generate a new range of disturbances (except for ω_1^j) and calculate $\tilde{V}_i^\pi \quad \forall\pi \in \Lambda$.

Letting $\mathcal{T}$ define a fixed number of iterations, we get the sample mean estimate $\psi(\pi) = \frac{1}{\mathcal{T}}\sum_{i=1}^{\mathcal{T}} V_i^\pi$ for each policy $\pi \in \Lambda$, which finally yields the approximation,

$$\tilde{V}_{H'-1}^{\pi^*}(f(s,\mathbf{a},\omega^j)) = \sum_{\pi\in\Lambda} \psi(\pi)\phi^{\mathcal{T}}(\pi) \tag{10}$$

We use (10) to derive the last term of our rollout expression in (7). That is, (10) is used for each of the subsequent states that are illustrated in Figure 3. The overall structure of the SAMW algorithm is presented in Algorithm 1, where we predefine $\mathcal{T}$ experimentally to ensure a limited runtime of the algorithm. Moreover, notice that all disturbances, ω_t^j, can be generated prior to the running of Algorithm 1 as will be elaborated in Section 3.2.2.

Algorithm 1 The Simulated Annealing Multiplicative Weights algorithm.

$\phi(\pi) \leftarrow 1/|\Lambda| \quad \forall\pi \in \Lambda$ ▷ Initialize the distribution
for $i = 1$ to $\mathcal{T}$ **do**
  $disturbances \leftarrow getNewDisturbances()$ ▷ New disturbances: $\omega_2^j, \omega_3^j, \ldots, \omega_{H'}^j$
  $\tilde{V}_i^\pi \leftarrow evaluate(disturbances, \pi) \quad \forall\pi \in \Lambda$
  $\phi(\pi) \leftarrow update(\phi(\pi), \tilde{V}_i^\pi) \quad \forall\pi \in \Lambda$ ▷ Update the distribution using (8)
end for
$\psi(\pi) \leftarrow average(\tilde{V}_i^\pi \quad \forall i) \quad \forall\pi \in \Lambda$
$\tilde{V}^{\pi^*} \leftarrow weightedAverage(\psi(\pi) \quad \forall\pi \in \Lambda, \phi(\pi) \quad \forall\pi \in \Lambda)$ ▷ Derive approximation using (10)
return $\tilde{V}^{\pi^*}$

3.2.2 The Search Procedure

Our approach for deriving an action, $\mathbf{a}$, from the current action space, A_s, is based on a Greedy Randomized Adaptive Search Procedure (GRASP) [24]. That is, we conduct an iterative search consisting of two levels: (1) A greedy randomized solution, followed by (2) a local search procedure. We use this approach due to the combinatorial and greedy cost structure of the problem, ensuring that any immediate greedy allocation of requests will result in a low-cost solution.

GRASP is generally known to perform well in various scheduling problems, as shown in the bibliography by Festa & Resende, 2002 [13], and has previously been employed to solve a surgical scheduling problem by Cartes & Medina, 2016 [6] where the proposed model performed adequately compared to the optimal solution. A generalized structure of the GRASP heuristic is presented in Algorithm 2.

Algorithm 2 Generalized structure of the GRASP heuristic.

```
while elapsedTime < maximumTime do
    a ← buildGreedyRandom(s)  ▷ Construct greedy randomized solution from current state s
    stop ← false
    while stop = false do
        a ← localSearch(a, s)  ▷ Try to improve the solution by local search
        stop ← checkStoppingCriteria()
    end while
end while
    return bestFound(a)  ▷ Return best solution from the entire search
```

For the greedy randomized solution, we generate a candidate list by enumerating all feasible allocations for each of the current requests, p_{ij}. Next, each of these allocations are ranked according to their apparent lowest cost increase. We then restrict this list to the $\alpha \in \mathbb{N}$ allocations with highest rank, and finally pick an allocation by random for insertion in the schedule, r_{ikl}. This process is conducted recursively until all requests have been allocated to the schedule.

For the ranking of each candidate allocation we conserve runtime for the later local search procedure, by only considering the current cost function, $C(s, \mathbf{a})$. Recall from (4) that the cost induced at every state is comprised of firstly a fixed setup cost, secondly an overtime-cost, and lastly a penalty for outsourcing requests. Thus, for an allocation to a room $k \in R$ on a day $l \in T$ we evaluate a candidate on the difference,

$$\Delta^i = o^i_{kl} - o^{i-1}_{kl} + (y^i_{kl} - y^{i-1}_{kl}) \cdot \kappa + q \cdot \phi \tag{11}$$

where Δ^i is the increase in cost *if* the i'th allocation is conducted for $i = 1, \ldots, \sum_{i,j \in P,J} p_{ij}$. Further, $o^i_{kl} \in \mathbb{R}_0$ and $y^i_{kl} \in \{0, 1\}$ is the overtime-cost and open-room indicator for the room $k \in R$ and day $l \in T$ for which the

request is allocated, similar to (4). Notice that we can consider the cost *on* allocation, so $o_{kl}^{i} \geq o_{kl}^{i-1}$ and $y_{kl}^{i} \geq y_{kl}^{i-1}$. In addition, $q \in \{0, 1\}$ indicates if the request is outsourced; and κ and ϕ is the fixed setup cost and outsource penalty, respectively. Lastly, o_{kl}^{0} and y_{kl}^{0} are inherited directly from the current state, s.

Afterwards, the local search procedure improves the solution that has been created on the greedy randomized level by intensifying the search. This is the only time in the search that the value function, $\tilde{V}_{H'-1}^{\pi}$, is taken into account. We base this level on a *first-best hill climber* using the evaluation function,

$$z = C(s, \mathbf{a}) + \frac{1}{N} \sum_{j=1}^{N} \tilde{V}_{H'-1}^{\pi}(f(s, \mathbf{a}, \omega^{j})) \tag{12}$$

based on the expression in (7). Our implementation of GRASP for the problem of searching for a suitable action $\mathbf{a} \in A_s$ is presented in Algorithm 3.

Algorithm 3 Our GRASP implementation in the search for an action $\mathbf{a} \in A_s$

$\mathbf{a}^* \leftarrow (0, 0, \cdots, 0)^T$
$z^* \leftarrow \infty$
$disturbances \leftarrow generate()$ ▷ Generate all required disturbances: ω_t^j
while $stillTimeLeft$ **do**
  $\mathbf{a} \leftarrow (0, 0, \cdots, 0)^T$
  for all $\sum_{i,j \in P,J} p_{ij}$ **do**
    $list \leftarrow getCandList(\mathbf{a}, s, \alpha)$
    $\mathbf{a} \leftarrow pickRandom(\mathbf{a}, list)$ ▷ Pick randomly from restricted candidate list
  end for
  $y \leftarrow 0$
  $\mathcal{N} \leftarrow getNeighborhood(\mathbf{a}, s)$
  while $y < noImpromentand$ $y < |\mathcal{N}|$ and $stillTimeLeft$ **do**
    $z, i \leftarrow evaluateNewRandom(\mathcal{N}, disturbances)$ ▷ Evaluate random element i from $\mathcal{N}$
    if $z < z^*$ **then**
      $z^* \leftarrow z$
      $\mathbf{a}^* \leftarrow update(\mathcal{N}, i)$
      $\mathcal{N} \leftarrow getNeighborhood(\mathbf{a}^*, s)$
      $y \leftarrow 0$
    else
      $y \leftarrow y + 1$
    end if
  end while
end while
return $\mathbf{a}^*$

Here, we construct the neighborhood, $\mathcal{N}$, from an enumeration of every feasible single move of a procedure to a new room or day along with all feasible swaps between two procedures. We terminate the local search procedure by using an upper bound on evaluations without improvement, or if the entire neighborhood has been evaluated.

To make all solutions to the action **a** comparable, the disturbances that are required for the SAMW algorithm, as well as for (12), are generated during the initialization of the algorithm. Furthermore, we reuse the $\mathcal{T}$ sequences, $\omega_2^j, \omega_3^j, \ldots, \omega_{H'}^j$ between each sample path j. So, accounting for the model horizon, H', the number of iterations in the SAMW algorithm, $\mathcal{T}$, and the N subsequent states, we require a total of $N + \mathcal{T} \cdot (H' - 1)$ randomly generated disturbances for the execution of Algorithm 3.

4 Implementation & Results

In this section, we demonstrate our simulation-based MDP based on data from a Danish hospital. We use data on patient arrivals and ward resources to estimate the occurrence of requests, procedure duration, and room availability.

In Section 4.1 we present the hospital case along with a number of assumptions related to our model implementation. Next, in Section 4.2 we present the parameter tuning, followed by Section 4.3 where our approach is compared to a range of myopic policies using simulation.

4.1 Case & Data Description

For the investigated hospital, requests occur according to $|P| = \mathbf{288}$ different types. The occurrence-process is further assumed to be Poisson, in accordance with Spratt et al., 2019 [26], and stationary with known parameters. Each request will be subject to an availability-pattern for which we assume that every successive period of five days has *at most* one day where the designated physician is unavailable. In addition, all patterns occur with equal probability. Furthermore, the procedure duration is random, but with known mean and variance.

Data for the ten most frequent types of requests, accounting for 52% of the total occurrence rate, are presented in Table 1.

We assume that all requests have to be allocated. However, if the hospital does not have sufficient capacity within the current planning horizon, then a minimum number of requests are allowed to be outsourced. The fixed planning horizon is set to $H = \mathbf{20}$ days within which the capacity on the number of open rooms depends on the weekday, as shown in Table 2. In total, the hospital has three different rooms at disposal for which the opening-hours results in a total time-capacity of $w_k = \mathbf{7.5}$ hours.

The planner further has to account for equipment compatibility between procedure types and rooms. The compatibility between procedures and rooms for the ten most frequent types are presented in Table 3, where 1 indicates that the procedure is compatible with the room; otherwise the indicator is 0. Between all allocated procedures we assume a fixed buffer time of $m = \mathbf{0.5}$ hours.

Type	Occurrences pr. day	Duration Mean (h)	Duration Variance (h^2)
Procedure A	0.57	2.26	2.15
Procedure B	0.49	2.26	2.05
Procedure C	0.14	1.44	1.41
Procedure D	0.11	2.91	2.99
Procedure E	0.10	1.25	1.01
Procedure F	0.07	1.64	1.25
Procedure G	0.07	1.75	1.37
Procedure H	0.06	2.69	2.30
Procedure I	0.06	2.56	2.12
Procedure J	0.06	1.65	1.40

Table 1: Occurrence rate, sample mean duration, and variance obtained for the ten most frequent types of requests. These account for about 52% of the total occurrence rate. The full list has been uploaded to the journal.

Weekday	Monday	Tuesday	Wednesday	Thursday	Friday
Limit on open rooms	1	2	2	2	2

Table 2: Upper limit on number of open operating rooms for each respective weekday.

Type	Room A	Room B	Room C
Procedure A	1	0	0
Procedure B	1	1	0
Procedure C	1	1	1
Procedure D	1	1	1
Procedure E	1	1	1
Procedure F	1	1	1
Procedure G	1	1	1
Procedure H	1	1	1
Procedure I	1	1	1
Procedure J	1	1	1

Table 3: Compatibility between procedure types and rooms. Shown for the ten most frequent types. Number 1 indicates that a procedure is compatible with a room; otherwise the indicator will be 0. The full list has been uploaded to the journal.

4.1.1 Model Implementation

In accordance with hospital data studied by Spratt et al., 2019 [26] and Strum et al., 2000 [28] we assume the capacity utilization of room $k \in R$ is distributed according to a log-normal distribution. This distribution has probability density function

$$p_k(\tau) = \frac{1}{\varrho_k \tau \sqrt{2\pi}} \cdot e^{\frac{-(\ln \tau - \gamma_k)^2}{2\varrho_k^2}} \tag{13}$$

where $\gamma_k = \ln(\mu_k^2/\sqrt{\sigma_k + \mu_k^2})$, $\varrho_k = \sqrt{\ln(\sigma_k/\mu_k^2 + 1)}$, μ_k is the sum of the expected durations for all procedures allocated to room $k \in R$, and σ_k^2 is the corresponding sum of their variances. In addition, we use a polynomial function to evaluate the cost of performing procedures in overtime δ, assuming that $f(0) = 0$. That is,

$$f(\delta) = b_1\delta^2 + b_2\delta \tag{14}$$

In practice the parameters b_1 and b_2 would be adjusted to attain the desired slope and relation to the payed overtime-costs, and the more intangible costs of stretching the procedure duration into overtime. Later, we will assess the result of adjusting these parameters on the performance of our model.

For the SAMW algorithm we employ two base-policies in the set Λ. These have been chosen to account for the uncertainties in the resulting costs and at the same time maintain a reasonably fast evaluation time. We will refer to these base-policies as:

1. The **Anticipative Increased Cost Policy** (AIP)
2. The **Anticipative Weighted Cost Policy** (AWP)

In both policies, the current requests, p_{ij}, are allocated to the schedule, r_{ikl}, according to their expected duration, $E[Y_i]$, in ascending order. Each request at a time, they evaluate all *feasible* room-day pairs, $k, l \in R, T$, within the planning horizon and allocate the requests based on the lowest *anticipative* cost. The latter is estimated differently in each of the policies.

1. The AIP estimates the increased cost similar to (11), but for the difference, $o_{kl}^i - o_{kl}^{i-1}$, accounts for the future procedures that have not appeared in the schedule, yet. Specifically, the total capacity utilization is estimated from μ_{kl} and σ_{kl} (cf. the distribution in (13)), where each parameter is a sum of the already allocated procedures and an estimate of the future procedures. Thus, prior to allocating the request, the AIP assumes that

$$\mu_{kl} = \eta_l \cdot E[Y_G] + \sum_{i \in P}(r_{ikl} \cdot E[Y_i]) + (\sum_{i \in P}(r_{ikl}) - 1) \cdot m \tag{15}$$

and

$$\sigma_{kl}^2 = \eta_l \cdot Var(Y_G) + \sum_{i \in P}(r_{ikl} \cdot Var(Y_i)) \tag{16}$$

for each feasible room-day pair, $k, l \in R, T$, where $E[Y_G] = \sum_{i \in P} E[Y_i] \cdot \lambda_i/\lambda_G$ is the global weighted average duration, $Var(Y_G) = \sum_{i \in P} Var(Y_i) \cdot \lambda_i/\lambda_G$ is the global weighted average variance, and $\lambda_G = \sum_{i \in P} \lambda_i$ is the global rate of occurrence. Lastly, $\eta_l \in \mathbb{R}_{>0}$ estimates the additional number of requests that day $l \in T$ will be subject to in the future. Thus,

$$\eta_l = \sum_{x=t+1}^{l} (\lambda_G/(H \cdot |R| - d_x)) \tag{17}$$

for $l \geq t+1$; otherwise $\eta_l = 0$. Further, $d_x \in \mathbb{N}_0$ is the number of room-day pairs that are closed (due to capacity depletion) within the horizon relative to day $x \in \mathbb{N}_{t+1 \leq x \leq l}$.

2. For the AWP policy, each allocation depends again on the requests that have not appeared in the schedule yet. However, the AWP is based on the notion that uncertainty should be employed as a "weight" rather than an estimate of the potential overtime-costs. Consider the difference $o^i_{kl} - o^{i-1}_{kl}$ from (11). This time o^{i-1}_{kl} is evaluated by merely summing over the known procedures in r_{ikl}, whereas the resulting overtime-cost, o^i_{kl}, is based on (15)-(17). However, we change (17) to $\eta_l = \nu \cdot \sum_{x=t+1}^{l}(\lambda_G/(H \cdot |R| - d_x))$, where $\nu \in \mathbb{R}_{>0}$ determines the "weight" of these uncertain requests, and is determined experimentally.

4.2 Adjusting the Parameters

The MDP model parameters were assessed and adjusted by applying the model to a simulation framework. That is, we simulated the arrival of requests and their resulting utilization of capacity in the system by generating pseduo-random numbers. In this simulation, we have assumed that requests occur according to a Poisson process, and that the total capacity utilization of any room is distributed according to a log-normal distribution as defined by (13).

We randomly generated three different sets of seeds covering a simulation period of 565 days, and then replicated each run of the simulation on each respective set twice. 365 days were used to burn-in the simulation for which we used the AIP policy to save runtime, leaving 200 days to assess the model performance of the MDP. On each respective day, the MDP was given a 20 minutes time-limit (cf. Algorithm 3). This limit was chosen to ensure that results resemble a practical setting.

Tests were conducted using three different levels for each respective parameter. The parameters that were subject for testing, and their levels, are presented in Table 4. The number of sampled paths, N, and the SAMW iterations, $\mathcal{T}$, were tested with interaction resulting in a total of $(3 \times 3 + 3 + 3) \times 2 \times 3 = 90$ simulations. The remaining parameters were adjusted during preliminary testing of the model.

For the cooling schedule, β_i, we tested both a fixed cooling parameter, such that β_i remained constant for all $\mathcal{T}$ iterations, and an exponential decreasing continuous function, $\beta_i = \beta(i) = 1 + C(\epsilon, \mathcal{T})^{-(i-1)}$, where $C(\epsilon, \mathcal{T}) = e^{-\frac{ln(-1+\epsilon)}{\mathcal{T}}}$, and ϵ defines the final cooling value after $\mathcal{T}$ iterations.

Lastly, for the overtime function in (3) we used a setting with $b_1 = 10$ and $b_2 = 4$, and a fixed setup-cost of $\kappa = 100$. The penalty for outsourcing requests was set to $\phi = 1 \cdot 10^6$.

#	Parameter	Level 1	2	3
1	SAMW Iterations ($\mathcal{T}$)	**20***	50	100
2	Cooling Schedule	$\beta_i = 2$ (fixed)*	$\epsilon = \mathbf{1.1}$	$\epsilon = 1.001$
3	Sampled Paths (N)	**5***	10	30
4	Model Horizon (H')	50	**150***	300

Table 4: Parameters that have been subject to simulation. Parameter 1 & 2 are related to the SAMW algorithm (Algorithm 1), and 3 & 4 to the rollout expression (7). The bold font indicates the preliminary setting of the parameters during the simulations, whereas the star indicates the setting employed in the experiments in Section 4.3.

We measured the performance of each setting on the cumulated cost (over the 200 day simulation period) of both the overtime- and setup-costs; and the penalty from outsourcing requests. The parameters were then compared in a dot-plot and on their respective correlations to the amount of cumulated value. Interestingly, the cooling schedule showed to be more effective when held constant at $\beta_i = 2$ and decreasing in performance as ϵ increases; hence when β_i decreases at a faster rate. The cooling schedule had a distinct effect on the performance, whereas the remaining parameters were more inconclusive. The effect from the number of SAMW iterations, $\mathcal{T}$, was almost negligible with respect to the cumulated value, whereas the number of sampled paths, N, and the model horizon, H', depended more on the specific set of seeds for the simulation.

As regards the value of ν in the AWP, we employed a hill climber heuristic where the average performance was recursively evaluated over ten different sets of seeds until convergence. This resulted in a final weight of $\nu = 16.969$.

4.3 Numerical Experiments

In this section, we apply our MDP model based on the results from the parameter tests, and compare the performance to a range of different policies. These include a policy that resembles the behavior of a "manual" planner, which we will refer to as the Manual Policy (MP). Next, we compare the MDP performance to a more advanced heuristic search procedure.

The MP is based on the following assumptions:

1. The expected duration of each procedure is known to the planner.
2. The planner is familiar with procedure variability, but the exact distribution nor spreading is not known. For this reason, a fraction of the available capacity is used as a buffer such that a new procedure is not allowed to *start* within this time-interval. We will test two different buffer levels in our numerical experimetns. To ensure the planner can utilize the remaining capacity we allow the buffer to be violated by a maximum of 10% of the total capacity.

3. The exact costs are unknown to the planner. For this reason, the planner will try to utilize the setup-cost for a new room as much as possible. Firstly, the requests are sorted in ascending order similar to the policies in Λ, and then allocated in sequence to the room-day pair that results in the least amount of excess capacity. If there are no feasible allocations for the room-day pairs that are already in use, the planner will allocate the request to the latest unused room-day pair such that this new room will be subject to as many future requests as possible.

Our experiments were conducted using simulations similar to the tests in Section 4.2. Thus, a period of 365 days were used to burn-in the simulation, and 200 days to assess the performance of the model. However, simulations were extended to eight different sets of seeds and replicated five times on each set. Besides testing the model on a range of different seeds, we varied the parameters in the overtime function (14) on four different levels, presented in Table 5. Later, we will refer to the overtime-cost settings using the conventions presented in this table. Again, the MDP runtime was fixed at 20 minutes for each day over the entire length of the simulation.

Our experiments in this section include the MDP, MP and the policies in Λ. For the MP, we decided to employ a capacity buffer of 20% which resembles the fraction that is most commonly used by our case hospital. Also, to test if higher room utilization leads to a better performance we included tests with a 10% buffer.

Reference	b_1	b_2
Low	10	4
Medium	100	10
High	300	100
Very High	10,000	500

Table 5: Parameter settings for the overtime function (14). All four levels are tested at each of the eight sets of seeds.

In order to compare the performance across the different combinations of seeds (and thereby the behavior of the requests generated) and overtime-costs, we standardized the cumulated cost, including the penalty for outsourcing, by employing the conversion

$$x_{ijk} = \frac{y_{ijk} - \min_{k \in \mathcal{K}_{ij}}\{y_{ijk}\}}{\max_{k \in \mathcal{K}_{ij}}\{y_{ijk}\} - \min_{k \in \mathcal{K}_{ij}}\{y_{ijk}\}} \tag{18}$$

where y_{ijk} is the resulting cost of simulation run $k \in \mathcal{K}_{ij}$ using seeds i and overtime-cost setting j. Thus, for the five replications of the MDP, the MP with both 10% and 20% capacity buffer; and the policies in Λ, $|\mathcal{K}_{ij}|$ includes 9 runs for each combination of i and j.

The results are presented in Table 6 showing the performance of each model and overtime-cost setting, presented as both the average and standard deviation standardized cost. Furthermore, average runtimes of each model are presented in Table 7.

Table 6 shows a distinct difference between the MDP and the remaining policies, measured on both the average and standard deviation performance. The difference is especially distinct between the MDP and the MP, regardless of the capacity buffer. Notice that the MP with 20% capacity buffer yields an average of 1.000 and a standard deviation of 0.000 for the first three overtime settings because this policy resulted in the highest cost across all eight sets of seeds.

Interestingly, the benefit of using the MDP increases as function of the overtime cost. Simultaneously, the difference between the policies decreases, resulting in quite indifferent performance at the highest overtime cost level. Otherwise, the MP performs substantially better with a 10% instead of a 20% capacity buffer. Still, the anticipative policies in Λ yield substantially lower costs than both MP settings, where AWP results in both lower average and standard deviation cost than all the remaining policies, except at the highest level. The relative difference between the average performance of the MDP and AWP is quite large, but given the runtimes in Table 7 which shows that AWP derives a decision in about 3 milliseconds, the latter might be a suitable choice in many settings.

	Average					Std. Deviation				
Overtime	MDP	MP 10%	MP 20%	AIP	AWP	MDP	MP 10%	MP 20%	AIP	AWP
Low	0.009	0.473	1.000	0.188	0.075	0.014	0.120	0.000	0.149	0.053
Medium	0.075	0.499	1.000	0.234	0.096	0.027	0.130	0.000	0.161	0.050
High	0.015	0.526	1.000	0.267	0.136	0.016	0.133	0.000	0.158	0.054
Very High	0.019	0.977	0.927	0.800	0.857	0.016	0.037	0.097	0.107	0.070

Table 6: Performance of the MDP compared to the MP with a capacity buffer of 10% and 20%; and the base-polices in Λ. The models are compared on their standardized cost.

	Average runtime (s)
MDP	1200.00
MP 10%	$2.47 \cdot 10^{-3}$
MP 20%	$1.97 \cdot 10^{-3}$
AIP	$2.79 \cdot 10^{-3}$
AWP	$2.92 \cdot 10^{-3}$

Table 7: The average runtime in seconds for the MDP, the MP, and the base-policies in Λ. Note that the MDP always has a fixed runtime of 1200 seconds (20 minutes).

We emphasize that performances of advanced approaches (such as the MDP) are only relevant if a computational setup can be introduced into the hospital operations, which is an obvious advantage of a manual approach. However, if this is possible, then we should consider how another well-known scheduling method compares to the MDP performance. This is the purpose of the following section.

4.3.1 Further Validation

In this section, we compare our MDP to a GRASP heuristic with a myopic cost-structure. That is, we re-used the basic algorithmic structure that was presented in Algorithm 2, but without the anticipative costs. Instead, during the local search, we evaluate the solution on the sum of the expected overtime-cost and the fixed setup-cost over the entire planning horizon. Thus,

$$C'(s,\mathbf{a}) = \sum_{k,l \in R,T \setminus \{t+H\}} o_{kl} + \sum_{k,l \in R,T \setminus \{t+H\}} (y^{s^*}_{kl} - y^{s}_{k,l+1}) \cdot \kappa + \sum_{i,j \in P,J} q_{ij} \cdot \phi \quad (19)$$

We chose to compare our MDP to the myopic GRASP due to its performance in other scheduling problems [13, 24], and because the method can be applied without excessive computer programming, which is an advantage to hospitals converting from manual to computational planning.

Just as in our previous experiments, we simulated the performance of the GRASP heuristic with 20 minutes of runtime, and replicated each run five times on each combination of seeds and overtime-cost setting.

The result of the simulations are presented in Table 8, showing the average and standard deviation performance for each model and overtime-cost setting. The average performance has further been depicted in Figure 4. Again, the models are compared on their standardized cost according to (18), but re-calculated to fit the only two models that are compared in this section.

	Average		Std. Deviation	
Overtime	MDP	GRASP	MDP	GRASP
Low	0.276	0.201	0.422	0.354
Medium	0.302	0.357	0.328	0.382
High	0.392	0.463	0.365	0.411
Very High	0.453	0.493	0.294	0.303

Table 8: Performance of the MDP compared to the GRASP heuristic. The models are compared on their standardized cost

Table 8 shows that the performance of the two models is much more equal compared to our previous experiments. In fact, the GRASP outperforms the MDP in both average and standard deviation cost when the overtime-cost is

set to "Low". This corresponds to the parameters $b_1 = 10$ and $b_2 = 4$ which is the setting that the MDP was adjusted for. However, as the cost of stretching procedures into overtime increases, so does the MDP performance resulting in lower average (cf. Figure 4) and standard deviation cost for the remaining overtime settings.

We should further emphasize that in the cheapest setting, the overtime-cost does not exceed the cost of opening a new room until about 3 hours into overtime, which is longer than the expected duration for most of the occurring requests in our data. This may not apply to many real hospital settings. In addition, these experiments were conducted for a reasonably short simulation of 200 days; hence, if the improvement of exploiting the rolling and overlapping nature of the problem is small, then such advantage might only show for much longer periods of simulation.

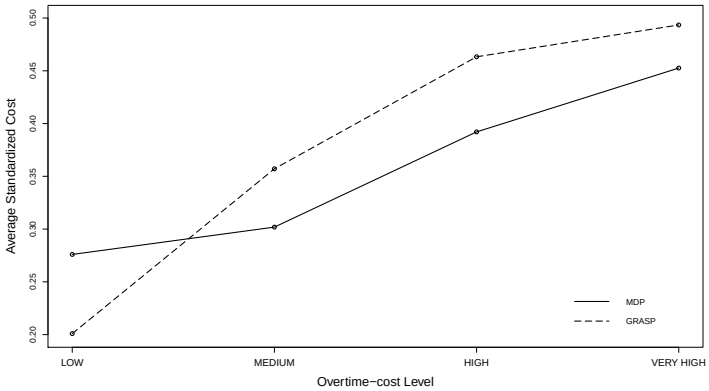

Fig. 4: Performance of the MDP and GRASP measured on their standardized cost. On average, the MDP yields better decisions when the overtime-cost is anything but at the cheapest level.

5 Conclusion

Aiming to apply and test a new approach to the problem of scheduling OTs, we developed a simulation-based MDP. The advantage of this type of modeling approach is that a sequence of decision problems are taken into account, which is often disregarded in OT planning.

Specifically, our approach derives a heuristic policy by evaluating a number of potential future scheduling paths from the currently observed state. This process is based on a predefined set of base-policies and employs an algorithm known as Simulated Annealing Multiplicative Weights (SAMW) [7]. We further consider that the state-dependent action space is intractable, and for

this reason we derive an action with a Greedy Randomized Adaptive Search Procedure (GRASP).

In order to validate our approach we conducted a number of numerical experiments based on simulation, and compared our results to both simple and more advanced myopic scheduling methods. Firstly, we validated a policy that resembles a manual planner. This indicated that a *distinct* improvement can be attained if our model is employed rather than scheduling requests manually. Furthermore, we found that a substantial improvement can be attained by employing a policy that accounts for future requests by weighting their contribution to the overtime-costs. We refer to this as the Anticipative Weighted Cost Policy (AWP). In addition, we found that a GRASP disregarding the rolling horizon performs only slightly worse than our MDP, and in fact better when the cost of stretching into overtime is sufficiently low.

5.1 Future Work

In future work more extensive numerical experiments should be considered. The difference in performance between the simulation-based MDP and myopic GRASP should be investigated by extending the period over which simulation is conducted, employing more levels on the overtime-cost setting, and more effective base-policies in Λ. Additionally, further work into more simple policies should be investigated to benefit the hospital cases where requests have to be allocated within a short time (e.g. below a few seconds).

Acknowledgments

This research was supported by the Danish governmental organization Region Sjælland. In particular, we would like to thank the department of Production, Research and Innovation for providing us with essential data and information on the operations of the Danish hospital operating theatres. In addition, we would like to thank Assistant Prof. Charlotte Vilhelmsen for providing us with insight into the literature on scheduling operating theatres and Professor Bo Friis Nielsen with statistical advice. This is a post-peer-review, pre-copyedit version of an article published in SN Operations Research Forum. The final authenticated version is available online at: http://dx.doi.org/10.1007/s43069-020-0009-6.

Statements

Funding: This study was funded by Region Sjælland.

Conflict of Interest: The authors declare that they have no conflict of interest.

References

1. S. Batun, B.T. Denton, T.R. Huschka, and A.J. Schaefer. Operating room pooling and parallel surgery processing under uncertainty. *INFORMS Journal on Computing*, 23(2):220–237, 2011. cited By 41.
2. D.P. Bertsekas and D.A. Castañon. Rollout algorithms for stochastic scheduling problems. *Journal of Heuristics*, 5(1):89–108, 1999. cited By 155.
3. J.T. Blake and M.W. Carter. Surgical process scheduling: a structured review. *Journal of the Society for Health Systems*, 5(3):17–30, 1997. cited By 59.
4. B. Cardoen, E. Demeulemeester, and J. Beliën. Optimizing a multiple objective surgical case sequencing problem. *International Journal of Production Economics*, 119(2):354–366, 2009. cited By 63.
5. B. Cardoen, E. Demeulemeester, and J. Beliën. Operating room planning and scheduling: A literature review. *European Journal of Operational Research*, 201(3):921–932, 2010.
6. Ignacio Cartes Rubilar and Rosa Medina Duran. A grasp algorithm for the elective surgeries scheduling problem in a chilean public hospital. *Ieee Latin America Transactions*, 14(5):7530430, 2333–2338, 2016.
7. H.S. Chang, M.C. Fu, J. Hu, and S.I. Marcus. An asymptotically efficient simulation-based algorithm for finite horizon stochastic dynamic programming. *IEEE Transactions on Automatic Control*, 52(1):89–94, 2007. cited By 12.
8. H.S. Chang, R. Givan, and E.K.P. Chong. Parallel rollout for online solution of partially observable markov decision processes. *Discrete Event Dynamic Systems: Theory and Applications*, 14(3):309–341, 2004. cited By 36.
9. B. Denton, J. Viapiano, and A. Vogl. Optimization of surgery sequencing and scheduling decisions under uncertainty. *Health Care Management Science*, 10(1):13–24, 2007. cited By 170.
10. B.T. Denton, A.S. Rahman, H. Nelson, and A.C. Bailey. Simulation of a multiple operating room surgical suite. pages 414–424, 2006. cited By 39.
11. E. Erdem, X. Qu, and J. Shi. Rescheduling of elective patients upon the arrival of emergency patients. *Decision Support Systems*, 54(1):551–563, 2012. cited By 10.
12. H. Fei, N. Meskens, and C. Chu. A planning and scheduling problem for an operating theatre using an open scheduling strategy. *Computers and Industrial Engineering*, 58(2):221–230, 2010. cited By 92.
13. P Festa and MGC Resende. Grasp: An annotated bibliography. *Operations Research/-computer Science Interfaces Series*, 15:325–367, 2002.
14. Organization for Economic Co-operation and Development (OECD). Oecd.stat - health care utilisation. Available at https://stats.oecd.org/Index.aspx?DataSetCode=HEALTH_STAT#.
15. F. Guerriero and R. Guido. Operational research in the management of the operating theatre: A survey. *Health Care Management Science*, 14(1):89–114, 2011.
16. H. Hansagi, B. Carlsson, and B. Brismar. The urgency of care need and patient satisfaction at a hospital emergency department. *Health Care Management Review*, 17(2):71–75, 1992. cited By 59.
17. M. Lamiri, X. Xie, A. Dolgui, and F. Grimaud. A stochastic model for operating room planning with elective and emergency demand for surgery. *European Journal of Operational Research*, 185(3):1026–1037, 2008. cited By 127.
18. J.H. May, W.E. Spangler, D.P. Strum, and L.G. Vargas. The surgical scheduling problem: Current research and future opportunities. *Production and Operations Management*, 20(3):392–405, 2011.
19. J.R. McMillan, M.S. Younger, and L.C. De Wine. Satisfaction with hospital emergency department as a function of patient triage. *Health Care Management Review*, 11(3):21–27, 1986. cited By 77.
20. D. Min and Y. Yih. Scheduling elective surgery under uncertainty and downstream capacity constraints. *European Journal of Operational Research*, 206(3):642–652, 2010. cited By 53.
21. National Audit Office of Denmark. Beretning til statsrevisorerne om hospitalernes brug af personaleresurser. Available at http://rigsrevisionen.dk/publikationer/2015/102014/.

22. Ministry of Health. Status paa sundhedsomraadet. Available at http://www.sum.dk/Aktuelt/Publikationer/Status-paa-sundhedsomraadet-sept-2015.aspx.
23. T.M. Range, D. Kozlowski, and N.C. Petersen. Dynamic job assignment: A column generation approach with an application to surgery allocation. *Discussion Papers on Business and Economics*.
24. Mauricio G.C. Resende and Celso C. Ribeiro. Grasp: Greedy randomized adaptive search procedures. *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques, Second Edition*, pages 287–312, 2014.
25. M. Samudra, C. Van Riet, E. Demeulemeester, B. Cardoen, N. Vansteenkiste, and F.E. Rademakers. Scheduling operating rooms: achievements, challenges and pitfalls. *Journal of Scheduling*, 19(5):493–525, 2016.
26. Belinda Spratt, Erhan Kozan, and Michael Sinnott. Analysis of uncertainty in the surgical department: durations, requests and cancellations. *Australian Health Review*, 43(6):706–711, 2019.
27. K. Steins, F. Persson, and M. Holmer. Increasing utilization in a hospital operating department using simulation modeling. *Simulation*, 86(8-9):463–480, 2010. cited By 23.
28. DP Strum, JH May, and LG Vargas. Modeling the uncertainty of surgical procedure times - comparison of log-normal and normal models. *Anesthesiology*, 92(4):1160–1167, 2000.
29. J.-S. Tancrez, B. Roland, J.-P. Cordier, and F. Riane. How stochasticity and emergencies disrupt the surgical schedule. *Studies in Computational Intelligence*, 189:221–239, 2009. cited By 3.
30. C. Van Huele and M. Vanhoucke. Analysis of the integration of the physician rostering problem and the surgery scheduling problem. *Journal of medical systems*, 38(6):43, 2014. cited By 4.
31. J.M. Van Oostrum, M. Van Houdenhoven, J.L. Hurink, E.W. Hans, G. Wullink, and G. Kazemier. A master surgical scheduling approach for cyclic scheduling in operating room departments. *OR Spectrum*, 30(2):355–374, 2008. cited By 83.
32. P.T. Vanberkel and J.T. Blake. A comprehensive simulation for wait time reduction and capacity planning applied in general surgery. *Health Care Management Science*, 10(4):373–385, 2007. cited By 74.
33. W. Xiang, J. Yin, and G. Lim. A short-term operating room surgery scheduling problem integrating multiple nurses roster constraints. *Artificial Intelligence in Medicine*, 63(2):91–106, 2015. cited By 2.
34. Jian Zhang, Mahjoub Dridi, and Abdellah El Moudni. A two-level optimization model for elective surgery scheduling with downstream capacity constraints. *European Journal of Operational Research*, 276(2):602–613, 2019.