OR Spectrum
DOI 10.1007/s00291-011-0270-3

REGULAR ARTICLE

The single-day surgery scheduling problem: sequential decision-making and threshold-based heuristics

William L. Herring · Jeffrey W. Herrmann

Abstract Scheduling elective surgeries is a dynamic, sequential decision-making process that must balance the costs of deferring waiting cases and blocking higher-priority cases. Although other surgery scheduling problems have received extensive treatment in the literature, this paper presents the first single-day scheduling problem formulation to capture this aspect of the scheduling process while also incorporating surgical block schedules, block release policies, and waiting lists. Theoretical results for the special case in which all cases have the same duration motivate a range of threshold-based heuristics for the general problem with multiple case durations. Our computational results demonstrate the effectiveness of the proposed heuristics and show how block release dates affect the quality of the scheduling decisions. Based on these results, we propose a new approach to surgery scheduling. In particular, to make more equitable waiting list decisions, operating room (OR) managers should gradually release unused OR time over the course of several days leading up to the day of surgery.

Keywords Healthcare · Surgery scheduling · Dynamic programming · Heuristics

1 Introduction

The health care system has been the focus of increasing attention from operations researchers and management scientists in recent years. The growth in this research

W. L. Herring (✉)
Applied Mathematics and Statistics and Scientific Computation Program,
University of Maryland, College Park, USA
e-mail: wherring@math.umd.edu

J. W. Herrmann
Department of Mechanical Engineering and Institute for Systems Research,
University of Maryland, College Park, USA

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

area is motivated by increasing costs and the rising demand for health care services and is facilitated by the improved quality and availability of data generated by the system. The problem of scheduling surgical patients into operating rooms benefits from a robust literature. This interest in surgery scheduling is intensified by the key role that operating room (OR) scheduling plays in determining hospital occupancy levels and because the OR is the most resource-intensive and profitable unit of a hospital (Macario et al. 1995; McManus et al. 2003). At its core, the surgery scheduling problem, in all its variations, involves the allocation of a fixed amount of resources (ORs, hospital staff) under uncertain demand (see Cardoen et al. 2010, for a thorough review). Like other scheduling problems, surgery scheduling approaches hope to make more efficient use of existing resources. This improved efficiency in the context of the health care system comes with the added societal benefit of lowering costs and increasing access to health care.

1.1 Literature review

The majority of hospitals schedule their OR suites using cyclic master, or block, surgery schedules, in which available OR space is assigned to specific surgical specialties, or service lines. For hospitals using block schedules, the literature on elective surgery scheduling describes the problem as consisting of three stages: (1) determining the amount of OR time to allocate to various surgical specialties, (2) creating a block schedule implementing the desired allocations, and (3) scheduling individual patients into available time (Blake and Donald 2002; Santibañez et al. 2007; Testi et al. 2007). First stage decisions typically reflect the long-term strategic goals of hospital management, such as meeting the demand for surgical specialties' services, achieving desired levels of patient throughput, or maximizing revenue (Blake and Carter 2002; Gupta 2007; Santibañez et al. 2007; Testi et al. 2007). The second and third stages represent medium- and short-term operational and tactical decisions, respectively, but differ markedly in their objectives. In recent years, block scheduling models have transitioned from simply implementing desired allocation levels (Blake and Donald 2002) to leveling hospital bed occupancy and minimizing overcapacity (Beliën and Demeulemeester 2007; van Oostrum et al. 2008). Research on individual patient scheduling, including patient selection, room placement, and sequencing, typically aims to minimize patient delays or maximize OR utilization (Denton et al. 2007; Guinet and Chaabane 2003). The large number of stakeholders involved in surgery scheduling has also motivated a number of multi-objective models for both block scheduling and individual patient scheduling (Blake and Carter 2002; Beliën et al. 2009; Cardoen et al. 2009).

A fundamental, but understudied, element of the day-to-day job of surgery scheduling is the transition from the generality of the block schedule (in which OR time is allocated to specialties) to the specificity of a completed schedule for a particular day (in which OR time is assigned to specific cases). As will be discussed below, individual patients are scheduled into blocks of OR time in a dynamic and sequential decision-making process. Block schedules are often incorporated as constraints in individual patient scheduling models (Hans et al. 2008; Pham and Klinkert 2008; Testi et al. 2007), but these models schedule patients all at once rather than sequentially over time. A key component of the schedule evolution is the scheduling of add-on cases

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

off of surgical waiting lists, but existing papers on this topic only consider decisions made at the tail end of the scheduling process and fail to capture earlier aspects of the process (Dexter et al. 1999; Dexter and Traub 2002; Gerchak et al. 1996). This paper proposes a new model for the single-day surgery scheduling problem that seeks to address these shortcomings.

The model proposed below has much in common with existing research on capacity allocation (Schütz and Kolisch 2010a,b) and, in particular, airline revenue management (Brumelle and Walczak 2003; Lee and Hersh 1993; Subramanian et al. 1999). In these problems, a finite resource (OR time, seats on a flight) with a fixed expiration date (the day of surgery, the departure time for the flight) must be allocated to competing demand classes. The demand from each of the classes arrives over time, and decision-makers must decide how much of the resource to allocate to lower priority classes and how much to reserve for higher priority classes. In these existing models, arriving demand must be accepted or rejected at the moment of its arrival and rejected demand is lost. In the surgery scheduling problem, however, lower priority demand is placed on a waiting list, or request queue, and can be accepted at a number of different decision points leading up to the day of surgery. While the proposed surgery scheduling model displays a threshold behavior similar to the revenue management models (particularly to Lee and Hersh 1993), the introduction of the request queue concept increases the complexity of the state space and complicates the analysis leading to these results.

The problem studied in the remainder of this paper is motivated by and models the dynamic nature of surgery scheduling, in which the OR schedule for a single day evolves over time. The immediate goal is to capture the most important relationships and develop insights into the process, particularly focusing on the often contentious nature of block release and RQ decisions. While the initial results presented in this paper are largely theoretical in nature, we believe that they lay an important foundation from which to design more equitable and transparent block release and RQ policies. Our modeling approach reflects our discussions with collaborators at the University of Maryland Medical Center (UMMC), whose response to our results and their policy implications has been positive. Their feedback informs a number of directions for future collaborative research that will be required before implementation at UMMC or any other particular hospital.

1.2 Dynamics of surgery scheduling

This description and the resulting problem statement are based on a case study of the surgery scheduling system at the UMMC (Herring 2011). Similar systems are used by other hospitals that use block scheduling (Dexter et al. 2003; Dexter and Macario 2004; Ozkarahan 2000).

Block schedules are concerned primarily with elective surgery, and the demand for elective surgery arrives over the course of multiple days before the day of surgery, rather than all at once. For a given day on which an OR suite is open for surgery, the available OR time is allocated to specific surgical specialties according to a block schedule. Blocks of OR time are initially controlled exclusively by these "primary" specialties, and the primary specialties are free to choose and sequence their cases ("primary" cases) within their allocated blocks as they see fit. Specialties or surgeons

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

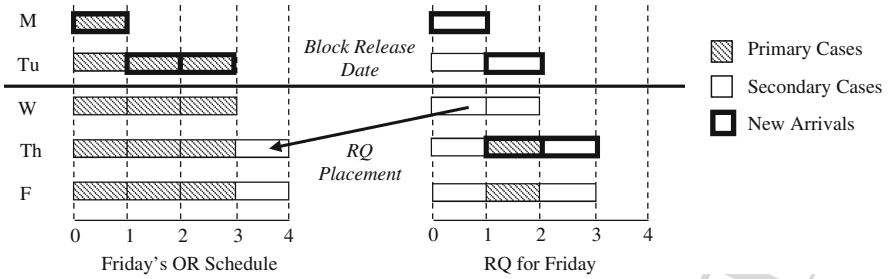

Fig. 1 The evolution of Friday's OR schedule and RQ for a single OR over the week before surgery

that do not have allocated OR time on that day, but still wish to perform a surgery, must submit their cases ("secondary" cases) to the surgical request queue (RQ) for that day. If a primary specialty's allocated OR time has been filled, it may also submit its excess cases to the RQ. In the period leading up to the day of surgery, cases accumulate on the RQ and OR managers look to schedule these RQ cases into OR time that has not been filled by the primary specialties. In practice, there is a set day before surgery, referred to as the *block release date*, after which OR managers can begin assigning RQ cases to unfilled blocks. Before the block release date, RQ cases may not be assigned to open times. After the block release date, the *request queue policy* determines how and when RQ cases will be assigned to open times. Together, the block release dates and request queue policies control the transition from the block schedule to the individual patient scheduling stage of surgery scheduling.

RQ decisions are made every day, once a day, before any new cases have arrived. In practice, OR managers must manage the schedule for many future surgery days, and it is impractical for them to spend time reconsidering the RQ cases too frequently. The demand for surgeries is often communicated to the hospital by the specialties once a day, typically after the surgeons have seen their patients in a clinic or made their rounds. It is also important to note that this problem does not consider the process of scheduling urgent (emergent) cases that are generated on the day of surgery. In practice, entire blocks may be reserved for urgent cases, or the schedule may include some slack for these cases. These and other similar approaches affect the scheduled capacity of the OR, which is taken as given in this problem.

The following example illustrates the importance of considering the sequential nature of the OR manager's decisions and how they interact with the demand for surgery. Figure 1 shows the evolution of the schedule for a single OR and the RQ for a given Friday over the course of a week before surgery, as demand for Friday's OR time arrives and the OR manager makes daily RQ decisions. The OR has been allocated to a primary specialty, and the demand for surgery is divided into primary cases and secondary cases. Note that not all cases arrive on the same day. If there is still open time on Friday's schedule, primary cases are added directly to the schedule; otherwise they are placed on the RQ. Secondary cases are always placed on the RQ. Because Wednesday is the block release date, RQ cases may not be added to Friday's schedule before Wednesday. Starting on Wednesday, the OR manager uses RQ policies to decide whether to schedule RQ cases into any time that is still available. In this

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

example, the decision on Wednesday morning to add a RQ case to the schedule means that there is no time available for the primary case that arrives on Wednesday, which is sent to the RQ (as seen on Thursday). Had the OR manager decided not to schedule the RQ case on Wednesday, then the primary case would have filled the OR, and the OR manager would have been prevented from scheduling the secondary case. In this dynamic, stochastic setting, there is no guarantee that the OR manager's decision will be optimal (when viewed ex post), but this paper aims to provide insights that can help the OR manager make better decisions on average.

The dynamic, sequential, and stochastic aspect of creating an OR schedule emphasizes the importance of considering the joint impact of block schedules, block release dates, and request queue policies. However, no existing work in the surgery scheduling literature addresses all three of these components simultaneously. As mentioned above, some recent papers use block schedules as constraints in models that schedule batches of individual patients (Hans et al. 2008; Pham and Klinkert 2008; Testi et al. 2007). A study of different block release dates concludes that the timing of the block release has little impact on OR efficiency (Dexter et al. 2003; Dexter and Macario 2004). However, these papers, which add single RQ cases to existing schedules at different points in time, fail to consider the potential effect of RQ decisions on the evolution of the schedule after the RQ case has been added. The other papers that consider scheduling add-on cases limit their analysis to the day before and the day of surgery, thus neglecting the role of the block release date (Dexter et al. 1999; Gerchak et al. 1996).

1.3 Problem statement

In our setting, both the demand for elective surgery and the RQ are day specific. Surgeons submit cases for a specific date in the future on which they would like to perform surgery. Similarly, cases on a given day's RQ that are never scheduled are not automatically rolled over to future days' RQs (although surgeons can resubmit their cases for future days). As a result, we can separate the general problem of scheduling the operating rooms into subproblems that consider only one day's schedule. Two types of costs are associated with the process of scheduling an OR suite that uses block scheduling: (1) utilization costs related to under- and over-utilization of the ORs and (2) customer satisfaction costs incurred when cases are forced to wait on the RQ or when primary cases are blocked from their allocated ORs due to RQ placements. The utilization costs are common in the literature, but the satisfaction costs are a distinctive feature of this formulation.

Consider again the example presented earlier. On Wednesday in Fig. 1, the OR schedule for Friday has room for one additional case and the RQ contains a case that is waiting to be scheduled. The OR manager has the following choice: either schedule the RQ case into the available time or defer scheduling the case (and consider it again on Thursday if the time is still open). Deferring the case leaves open the possibility that a primary case may arrive to use the remaining time, but is undesirable because the surgeon and the patient associated with the case must wait at least one more day for a decision. This satisfaction cost is defined as a *deferral cost*. If the RQ case is scheduled (thus filling the OR) and another primary case arrives, then the OR manager

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

has blocked the primary service line's access to its allocated room. This satisfaction cost is defined as a *blocking cost*. The balance between deferral costs and potential blocking costs informs the OR manager's decision to schedule or defer RQ cases. While these deferral and blocking costs are primarily satisfaction costs, it is important to note that their relative values may reflect a range of practical concerns, including differences in patient acuity levels, specialized resource needs, and contributions to revenue between surgical specialties.

On each day leading up to the day of surgery, the OR manager must choose the number of RQ cases to add to each OR's schedule for the day of surgery. The manager's objective is to minimize the expected total cost of deferral and blocking penalties incurred on the days before surgery and OR underutilization on the day of surgery. Block release dates restrict the days on which RQ cases can be added to the schedule and may differ from room to room. RQ policies dictate which decisions to make in which scenarios.

The dynamic relationship between decisions and costs suggests a stochastic dynamic programming (SDP) formulation. Because block release dates serve as constraints, the optimal policy combination would use no block release dates and use the optimal decisions from the unconstrained SDP as the RQ policy. For this reason, the formulation and analysis that follow consider an SDP unconstrained by block release dates. Because block release dates are used in practice (to protect OR time allocated to primary service lines and to reduce the OR manager's workload by limiting the number of days that must be considered), the added cost of imposing block release dates will be considered.

A special case of the single-day surgery scheduling problem for a single OR in which all of the cases have the same duration is formulated as an SDP and analyzed in Herring (2010). A more general formulation for a single OR with multiple case durations is presented in Sect. 2. Section 3 summarizes the results of the special case and analyzes the optimal solutions for the general case. Insights from these results motivate a range of threshold-based heuristics. Section 4 presents the results of computational tests that compare these heuristics and study the impact of block release dates. Finally, Sect. 5 offers some concluding remarks and indicates directions for future research.

Overall, the results in this paper demonstrate the effectiveness of the proposed threshold-based heuristics and show how block release dates affect the quality of the scheduling decisions. These results provide valuable insights into how OR managers can use threshold-based decision rules to make equitable waiting list decisions. Moreover, it is the first step towards modeling, understanding, and optimizing more realistic problems involving more case types, multiple operating rooms, and other complexities that are necessary in order to provide meaningful decision support to an OR manager.

2 Stochastic dynamic programming formulation

This section presents a stochastic dynamic programming formulation of the single-day surgery scheduling problem (SDSSP) that considers the general case with multiple case durations. The problem focuses on the development of the schedule for a single OR on a single, fixed day of surgery. The OR has been allocated to a primary specialty

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

according to a block schedule. The two sources of demand (primary and secondary) are assumed to be independent, as are the number of cases that arrive each day. RQ decisions for the OR must be made once a day on each day leading up to the day of surgery, and are made before any additional cases for the day of surgery arrive.

For modeling purposes, cases will be separated into different types based on their duration (the amount of OR time allocated to the case). While this approach requires discretizing the case durations, which may not capture the entire range of possible durations, there is a precedent in the literature for treating case durations in this manner (see, for instance, Guinet and Chaabane 2003). In addition to helping maintain the computational tractability of the SDP, this approach allows the analysis to stay focused on the principal issue of how OR managers balance potential blocking and deferral costs.

2.1 Formulation

The input data is defined as follows:

N = number of days before the day of surgery on which surgical demand is generated
C = capacity of the OR in hours
K = number of case types
d_k = duration (in hours) of cases of type k, for $k = 1, \ldots, K$

The stochastic demand for surgery is given by two classes of random variables associated with the primary cases and secondary cases. From this point forward, all references to days refer to the number of days before the day of surgery, with day 0 representing the day of surgery.

T_j^k = number of primary cases of type k that arrive on day j, $j = 1, \ldots, N$
R_j^k = number of secondary cases of type k that arrive on day j, $j = 1, \ldots, N$

The blocking and deferral costs for each day before surgery and the utilization costs for the day of surgery are similarly defined.

h_j^k = deferral cost on day j for cases of type k, $j = 1, \ldots, N$
r_j^k = blocking cost on day j for cases of type k, $j = 1, \ldots, N$
h_0^k = penalty for unscheduled cases of type k left on RQ on the day of surgery
r_0 = penalty for unused space in the OR on the day of surgery

Three classes of state variables are required to represent the state of the OR schedule at the start of day j (before the day j's RQ decisions have been made and before any new cases have arrived). The number of blocking eligible hours in the OR reflects the presence of RQ cases that were added to the schedule on previous days but have not yet incurred a blocking penalty. This auxiliary state accounts for the fact that a primary case might be blocked by a RQ case scheduled on a previous day.

C_j = available hours remaining on the OR schedule on day j, $j = 0, \ldots, N$
B_j = number of blocking eligible hours on the OR schedule on day j, $j = 0, \ldots, N$
W_j^k = number of cases of type k on the RQ on day j, $j = 0, \ldots, N$

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

Finally, the decision variables can be defined.

x_j^k = number of RQ cases of type k to add to the OR schedule on day j, $j = 0, \ldots, N$

With the exception of the day of surgery, the costs incurred each day are separated into deferral costs and blocking costs. Deferral penalties are assessed only up to the number of RQ placements that are feasible.

$$ND_j^k = \text{number of cases of type } k \text{ deferred on day } j,\ j = 1, \ldots, N$$
$$= \min\left(\left\lfloor \frac{C_j}{d_k} \right\rfloor, W_j^k\right) - x_j^k$$

When the RQ decisions are made on day j, that day's arrivals have not yet occurred. Therefore, the number of blocking penalties incurred on day j is a random variable that depends upon the value of the decision variables and on the arrival of primary cases. The consideration of different case durations raises an important question related to computing blocking costs. If a longer primary case is blocked and sent to the RQ, should a blocking penalty be assessed for the entire case or only for the part of the case that overlaps with the blocking eligible hours on the OR schedule? Suppose, for example, that the OR schedule has one available hour ($C_j = 1$) and one blocking eligible hour ($B_j = 1$) and that a primary case with a duration of 2 h arrives ($T_j^k = 1$ for the case type with $d_k = 2$). Because this case will not fit in the available time, it is blocked and placed on the RQ. The blocking penalty can be assessed either for the entire 2-h duration or for just the fraction of the case that overlaps with the blocking eligible hour. (Both approaches will be considered during computational testing). Computing these blocking quantities requires an algorithm (as opposed to a closed form expression).The pseudocode for this algorithm, referred to as *ProcessDay*, is given in the Appendix.

NB_j^k = number of cases of type k blocked on day j, $j = 1, \ldots, N$

FB_j^k = fraction of cases of type k blocked on day j, $j = 1, \ldots, N$

fit_j^k = number of primary cases of type k added to the OR schedule on day j, $j = 1, \ldots, N$

This is now sufficient information to define the value function, Bellman's optimality equation, and the day-to-day transition equations. Note that the daily blocking penalty is a convex combination of the number blocked and the fraction blocked, with the weighting parameter λ satisfying $0 \leq \lambda \leq 1$. This parameter allows us to address the question of full or partial blocking, and its impact will be explored during computational testing. For convenience, some quantities are expressed in vector form in the optimality equation and subsequent discussion.

For $j = 1, \ldots, N$:

$$V_j\left(C_j, B_j, \boldsymbol{W}_j\right) = \text{minimum expected remaining cost from state } \left(C_j, B_j, \boldsymbol{W}_j\right) \text{ on day } j$$
$$= \min_{x_j}\left\{\sum_{k=1}^{K}\left(h_j^k N D_j^k + r_j^k E\left[\lambda N B_j^k + (1-\lambda) FB_j^k\right]\right)\right.$$
$$\left. + E\left[V_{j-1}\left(C_{j-1}, B_{j-1}, \boldsymbol{W}_{j-1}\right)\right]\right\}$$

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

$$\text{s.t.} \quad \sum_{k=1}^{K} d_k x_j^k \le C_j$$

$$0 \le x_j^k \le W_j^k \quad \text{for } k = 1, \ldots, K$$

$$\text{where} \quad C_{j-1} = C_j - \sum_{k=1}^{K} d_k \left(x_j^k + fit_j^k \right)$$

$$B_{j-1} = B_j + \sum_{k=1}^{K} d_k \left(x_j^k - FB_j^k \right)$$

$$W_{j-1}^k = W_j^k - x_j^k + R_j^k + \left(T_j^k - fit_j^k \right)$$

The boundaries for the SDP are day N and day 0. On day N the system is initialized to empty, and on day 0 the boundary costs come from the unused time on the OR schedule and the cases that remain on the RQ. As with deferral costs, the objective function on day 0 penalizes only those cases on the RQ that could have been feasibly added to the schedule. The boundary value function is defined as follows:

$$V_0\left(C_0, B_0, \boldsymbol{W}_0\right) = \min_{x_0} \left\{ \sum_{k=1}^{K} h_0^k \left(\min\left(\left\lfloor \frac{C_0}{d_k} \right\rfloor, W_0^k \right) - x_0^k \right) + r_0 \left(C_0 - \sum_{k=1}^{K} d_k x_0^k \right) \right\}$$

$$\text{s.t.} \sum_{k=1}^{K} d_k x_0^k \le C_0$$

$$0 \le x_0^k \le W_0^k \quad \text{for } k = 1, \ldots, K$$

Some minor rearrangements of the day 0 value function show it to be an integer knapsack problem with some additional constants (with respect to $\mathbf{x_0}$) in the objective function.

$$V_0\left(C_0, B_0, \boldsymbol{W}_0\right) = r_0 C_0 + \sum_{k=1}^{K} h_0^k \min\left(\left\lfloor \frac{C_0}{d_k} \right\rfloor, W_0^k \right) - \min_{x_0} \left\{ \sum_{k=1}^{K} \left(h_0^k + d_k r_0 \right) x_0^k \right\}$$

$$= r_0 C_0 + \sum_{k=1}^{K} h_0^k \min\left(\left\lfloor \frac{C_0}{d_k} \right\rfloor, W_0^k \right) + \max_{x_0} \left\{ \sum_{k=1}^{K} \left(h_0^k + d_k r_0 \right) x_0^k \right\}$$

The final details needed to complete the formulation involve some aspects of the *ProcessDay* algorithm for computing the blocking quantities. The algorithm represents

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

how the system processes a particular realization of the primary case arrival random variables(see the Appendix for the pseudocode). In order to process arrivals of different types, it is necessary to assume some sort of case type prioritization on behalf of the primary service line. Because this prioritization is beyond the control of the OR manager, it is not included as a decision variable in the model. (Note that the prioritization of the RQ case types is within the OR manager's control and is reflected as such in the daily RQ decision variables). We assume throughout that the primary service line prioritizes its cases by decreasing duration. That is, longer duration cases receive priority over the shorter duration cases. Other prioritization rules could easily be considered within the framework of the algorithm.

2.2 Implementation issues and solution structure

As the problem is currently formulated, there is no upper bound on the size of the request queue. In order to implement a solution algorithm for the SDP, however, the RQ must be truncated at some point. Early implementations of code for the SDP showed that a truncation rule that combined the states with large RQs by ignoring any values beyond the point of truncation created a "bounce-back" effect that impacted the optimal solutions. Apparent irregularities in the optimal solutions followed the location of the truncation, necessitating a more sophisticated technique for limiting the number of RQ states that would not allow the system to prematurely return from the point of truncation.

Instead of truncating, we combined the states with large RQs (those greater than a given boundary) into a state in which the RQ is described as "infinite." In order to justify that this technique does not produce solution irregularities of its own, two properties must hold. First, once the RQ state variable for a case type exceeds what will fit into the remaining time on the schedule, this condition must continue to hold for all subsequent states. Second, the optimal decisions must be identical for all states beyond the selected boundary. We claim below that these properties hold if the RQ boundary for each case type is beyond the range of feasible decisions, which allows the RQ boundaries to be set as small as possible and greatly reduces the computational complexity of the SDP implementation.

The following definitions will be useful both in the statement and proof of the desired claims. The first definition applies to $j = 1, \ldots, N$

$$F_j\left(C_j, B_j, \boldsymbol{W_j}, \boldsymbol{x}_j\right) = \sum_{k=1}^{K}\left(h_j^k N D_j^k + r_j E\left[\lambda N B_j^k + (1-\lambda) F B_j^k\right]\right) + E\left[V_{j-1}\left(C_{j-1}, B_{j-1}, \boldsymbol{W}_{j-1}\right)\right]$$

$$F_0\left(C_0, B_0, \boldsymbol{W_0}, \boldsymbol{x_0}\right) = \sum_{k=1}^{K} h_0^k\left(\min\left(\left\lfloor \frac{C_0}{d_k}\right\rfloor, W_0^k\right) - x_0^k\right) + r_0\left(C_0 - \sum_{k=1}^{K} d_k x_0^k\right)$$

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

The dependence on the decision variables, x_j^k, is not explicitly stated on the right hand side of the first definition but is implied for all quantities that depend on x_j^k. With these functions thus defined, an alternative statement of the value function for $j = 0, \ldots N$ can be given, with the minimization still subject to the same constraints described above. This re-statement of the value function also provides a concise way to refer to the optimal decision for each feasible state.

$$V_j\left(C_j, B_j, \boldsymbol{W}_j\right) = \min_{xj}\left\{F_j\left(C_j, B_j, \boldsymbol{W_j}, \boldsymbol{x_j}\right)\right\}$$
$$\boldsymbol{x}_j\left(C_j, B_j, \boldsymbol{W}_j\right) = \arg\min_{\boldsymbol{xj}}\left\{F_j\left(C_j, B_j, \boldsymbol{W_j}, \boldsymbol{x_j}\right)\right\}$$

Finally, a quantity referred to as the "trimmed RQ state" is defined for each feasible state that reflects the maximum number of feasible RQ placements for each case type.

$$\hat{W}_j^k = \min\left(\left\lfloor \frac{C_j}{d_k} \right\rfloor, W_j^k\right)$$

The two claims below represent the two properties described above. The first claim justifies the notion that once the number of RQ cases exceeds the remaining capacity, it will continue to do so. The second claim states that the value functions and optimal decision vectors are identical for all states where the number of a certain case type on the RQ exceeds what is feasible. Proofs of these claims are given in the Appendix.

Claim 1 *If* $W_j^k \geq \left\lfloor \frac{C_j}{d_k} \right\rfloor$, *then* $W_{j-1}^k \geq \left\lfloor \frac{C_{j-1}}{d_k} \right\rfloor$ *for all possible future states.*

Claim 2 *For all feasible states* $\left(C_j, B_j, \boldsymbol{W}_j\right)$ *for all days* $j = 0, \ldots, N$, *the following equalities hold:*

(i) $F_j\left(C_j, B_j, \boldsymbol{W}_j, \boldsymbol{x}_j\right) = F_j\left(C_j, B_j, \widehat{\boldsymbol{W}}_j, \boldsymbol{x}_j\right)$
(ii) $V_j\left(C_j, B_j, \boldsymbol{W}_j\right) = V_j\left(C_j, B_j, \widehat{\boldsymbol{W}}_j\right)$
(iii) $\boldsymbol{x_j}\left(C_j, B_j, \boldsymbol{W}_j\right) = x_j\left(C_j, B_j, \widehat{\boldsymbol{W}}_j\right)$

The final two parts of Claim 2 are the justification for combining all states past the RQ boundary into one state. For case type k, $\left\lfloor \frac{C}{d_k} \right\rfloor$ is used as the boundary because it equals the maximum number of cases of type k that will fit into an empty OR schedule. Because the value function and optimal decisions are identical for all states beyond this boundary (all other things being equal), aggregating these states will not impact the resulting optimal policy.

3 Optimal solution behavior

3.1 The special case of unit durations

In the special case when all cases have a unit duration (SDSSP1), the optimal decisions follow a threshold policy (Herring 2010). That is, for each day before the day of surgery, a fixed number of hours are reserved for future primary cases, and the optimal decisions

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

Table 1 Optimal thresholds for SDSSP1 across a range of cost ratios and arrival scenarios

Blocking-to-deferral cost ratio	Arrival scenarios														
	Early					Middle					Late				
	Days before surgery														
	4	3	2	1	0	4	3	2	1	0	4	3	2	1	0
1:1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3:1	3	1	1	1	0	2	2	2	1	0	1	2	2	2	0
5:1	4	2	1	1	0	3	3	2	2	0	3	3	3	3	0
7:1	5	3	2	1	0	4	4	3	2	0	4	4	4	4	0

take the system as close to this threshold as possible. Furthermore, the optimal thresholds can be computed efficiently (without solving the SDP) from the daily deferral and blocking costs and the primary case arrival distributions. The existence and optimality of these thresholds are formally stated in the following claim (proved in Herring 2010).

Claim 3 *Assume that all cases have unit duration and the daily costs satisfy* $h_j \leq r_j$ *for all* $j \geq 0$ *and* $r_{j+1} \geq r_j$ *for all* $j \geq 1$. *Then for all* $j \geq 0$ *there exists a threshold* Y_j *such that the optimal decision for all feasible day* j *states* (C_j, B_j, W_j) *is given by min* $(W_j, \max(0, C_j - Y_j))$.

In addition to the threshold structure, two other observations stand out in this expression for the optimal decision. First, the blocking eligible state (B_j) plays no role in the optimal decision. This state essentially keeps track of the schedule's RQ history, thus implying that past RQ decisions should have no bearing on the current optimal decision. Second, the number of cases on the RQ (W_j) influences the optimal decision only as an upper bound on the number of RQ cases that can be scheduled. This indicates that having a very large number of cases on the RQ should not force the OR manager to place more of them.

Table 1 shows the optimal thresholds for SDSSP1 across a range of different input scenarios for an OR schedule that has a capacity of four cases (for the complete input data, see Herring 2010). Recall that the thresholds reflect how much space to save on the OR schedule for future primary cases, so higher thresholds lead to fewer RQ cases being scheduled. The behavior of the optimal thresholds is fairly intuitive. When blocking and deferral costs hold equal weight, the result is a greedy RQ policy. However, as blocking becomes more expensive (relative to deferring), fewer RQ cases are scheduled. The thresholds are also highly dependent on the timing of the primary case arrivals. More RQ cases are scheduled after the bulk of the expected demand has passed, but when primary cases are expected to arrive late, few if any RQ cases are accepted until the day of surgery.

3.2 The general case with multiple durations

The simple and intuitive nature of the optimal thresholds for SDSSP1 raises a fundamental question about the extent to which this behavior is apparent in the general

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

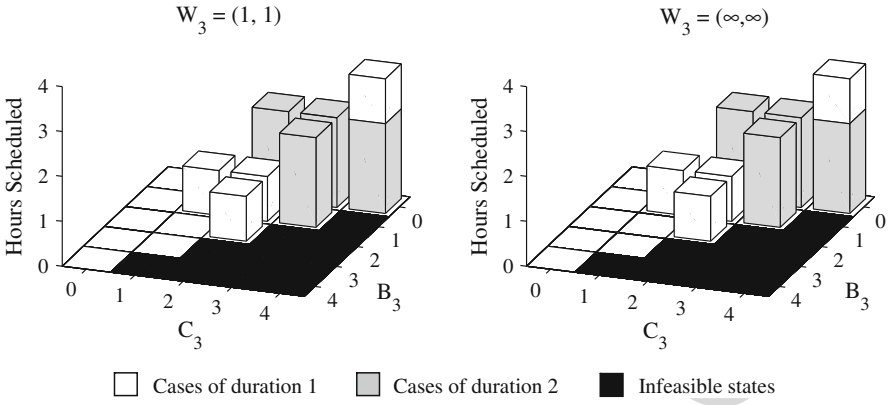

Fig. 2 Policy maps showing the number of hours of cases scheduled by the optimal decisions 3 days before surgery for two RQ states across all feasible hours remaining (C_3) and blocking eligible hours (B_3) states

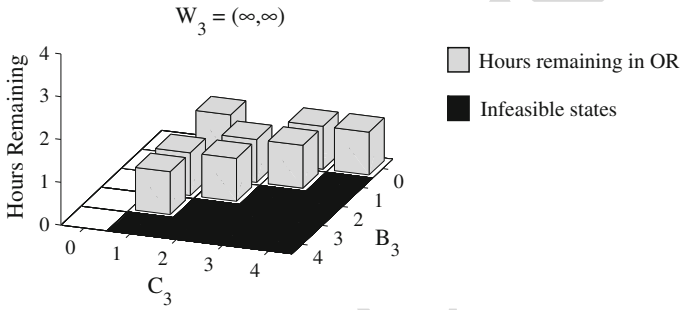

Fig. 3 Hours remaining on the OR schedule after the optimal decisions 3 days before surgery for a single RQ state across all feasible hours remaining (C_3) and blocking eligible hours (B_3) states

case. The policy maps in Fig. 2 show the optimal decisions for a single instance of SDSSP across a representative selection of states. In this example, the scheduled OR capacity is 4 h, and there are two case types: 1-h cases and 2-h cases. The rest of the input data come from one of the test problems described in the next subsection. The policy maps show the total hours of each case type taken from the RQ and placed on the OR schedule by the optimal decision for each state. A single map shows all possible hours remaining (C_j) and blocking eligible hours (B_j) states for a single RQ state on a particular day. Note that the sum of the hours remaining and the blocking eligible hours cannot exceed the scheduled capacity of the room (as indicated by the infeasible states).

The fact that the policy map does not change when the number of cases on the RQ is increased confirms the notion (as indicated by Claims 1 and 2 in Sect. 2) that having more cases on the RQ does not pressure the optimal decision into taking more cases. The maps also immediately show that certain behaviors from SDSSP1 do not extend to the general case. In particular, a change in the number of blocking eligible hours can change the optimal decision. This change is evident between $B_3 = 0$ and $B_3 = 1$ for both maps. Figure 3 depicts the number of available hours on the OR schedule after

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

Table 2 Input data for a set of 162 test problems for the general case of SDSSP

Parameter	Notation	Range of values
Blocking parameter	λ	0, 0.5, 1
Deferral cost ratio	$h_j^2 : h_j^1$	1:1, 2:1
Blocking cost ratio	$r_j^2 : r_j^1$	1:1, 2:1, 3:1
Blocking-to-deferral cost ratio	$r_j^1 : h_j^1$	1:1, 3:1, 5:1
Case type arrival rate ratio	$E[T_j^2] : E[T_j^1]$	1:2, 1:1, 2:1

the optimal decisions and clearly illustrates the fact that the decisions do not exactly follow a threshold pattern. However, it does appear that the decisions are guided by a "target" threshold. In this case, the OR manager generally seeks to leave 1 h available in the OR, and deviates from this target only by small amounts and in a few situations.

3.3 Test problems

The results in Table 1 show that the optimal thresholds for SDSSP1 depend on the primary case arrival distribution and on the ratio of blocking costs to deferral costs. Two other potential factors merit exploration: the ratio of the deferral costs and blocking costs between the case types and the weighting parameter (λ) that balances the number blocked $\left(NB_j^k\right)$ and the fraction blocked $\left(FB_j^k\right)$.

Table 2 shows the ranges of the input data used to create an initial set of 162 test problems. The remaining problem parameters are held constant for all the instances. The scheduled capacity of the OR is set to 4 h ($C = 4$), and there are two case types with durations of 1 and 2 h, respectively ($d_1 = 1, d_2 = 2$). The problems are solved over a period of 4 days before the day of surgery ($N = 4$). The blocking and deferral costs are held constant from day to day for each instance (following the ratios in Table 2), and the cost of unused OR time is set to ten ($r_0 = 10$). The case arrivals on each day follow Poisson distributions, and a total of 6 h of primary cases and 6 h of secondary cases are expected over the time horizon. For the secondary cases, an equal number of each case type is expected $\left(E\left[R_j^1\right] = E\left[R_j^2\right] = 0.5\right)$, and the expected number of primary cases is weighted between the two case types as indicated by the ratios in Table 2. The expected number of primary arrivals is the same every day, implying that this initial set of test problems will not capture the influence of the arrival timing on the optimal solutions. This factor and the impact of block release dates will be explored in Sect. 4.

3.4 Solution behavior

A thorough analysis of policy maps similar to those in Figs. 2 and 3 for each of the test problems reveals insight into three fundamental aspects of the solution behavior: (1) the extent to which the solutions follow an approximate threshold pattern, (2) the nature of the target thresholds (relative to the input data), and (3) the factors that drive

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

the selection of one case type over another. While it is not feasible to present the entire set of policy maps here (see Herring 2011 for a representative selection), the results of the analysis can be summarized as follows.

The most striking and important trend is that the optimal decisions exhibit an approximate threshold behavior like the example shown in Figs. 2 and 3. In many cases the thresholds are exact, but even when the optimal solutions deviate from the apparent threshold they do so in only a small number of states and rarely by more than a single hour in either direction.

Of the five factors listed in Table 2, two have no significant impact on the nature of the optimal decisions and apparent target thresholds. First, the impact of the blocking weight parameter (λ) is minimal. Smaller λ values effectively lower the blocking costs, and, as a result, more hours of RQ cases tend to be scheduled when $\lambda = 0$ than when $\lambda = 1$. However, the change rarely amounts to more than an hour's difference for a handful of states and never affects the apparent threshold guiding the optimal decisions. Similarly, the relative prevalence of one primary case type over another $\left(E\left[T_j^2\right] : E\left[T_j^1\right]\right)$ has very little influence on the optimal decisions. This suggests that the total hours of cases expected each day (which is held constant throughout) is more important in determining the target threshold than the prevalence of a particular case type. Finally, neither factor has any impact on the optimal solutions' tendency to prefer one case type over another (i.e. taking two 1-h cases over one 2-h case, or vice versa).

These observations leave the deferral and blocking cost structure as the primary determinant of the target threshold and the selection between the two case types. As in SDSSP1, the ratio of blocking cost to deferral cost (in the form of the $r_j^1 : h_j^1$ ratio) has the most influence on the approximate thresholds for the general SDSSP. Furthermore, the apparent thresholds for the three ratios listed in Table 2 (1:1, or *greedy*; 3:1, or *low*; and 5:1, or *high*) exhibit threshold trends much like those observed in Table 1 for SDSSP1. When the two costs are weighted equally, the optimal decisions universally try to fill the OR schedule as much as possible (thus the 'greedy' label). When the ratio is 'low,' the thresholds roughly aim to preserve enough open time for the total hours of cases expected each day. When the ratio is 'high,' the thresholds move to preserve enough open time for the total cumulative remaining hours of expected cases.

The other cost ratios included in this initial set of test problems (the blocking and deferral cost ratios between the case types) also impact the optimal solution behavior. Recall that the cost ratio discussed above is set for the first type $\left(r_j^1 : h_j^1\right)$, but not for the second case type $\left(r_j^2 : h_j^2\right)$. However, the between-case-type ratios ($h_j^2 : h_j^1$ and $r_j^2 : r_j^1$) affect the ratio of the blocking cost to the deferral cost for the second case type. When the between-case-type ratios combine to the make the cost ratio for the second case type higher than the cost ratio for the first case type, the approximate thresholds increase by as much as an hour. This pattern is reversed when the between-case-type ratios combine to make the cost ratio lower for the second case type. While these trends in the target thresholds are perhaps not as clean as those observed for SDSSP1, they provide evidence that the optimal solutions to SDSSP2 exhibit threshold behavior similar to that observed for SDSSP1.

In some cases, the optimal solutions prefer one case type over another. In particular, when the deferral costs are the same for both case types $\left(h_j^2 : h_j^1 = 1 : 1\right)$, the

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

solutions uniformly prefer the shorter case type. When the deferral costs are proportional to the case durations $\left(h_j^2 : h_j^1 = 2 : 1\right)$, the prioritization becomes more difficult to decipher. However, a weak preference seems to exist for the type with the higher ratio of blocking cost to duration. Together, these observations suggest a prioritization that first considers the ratio of deferral cost to duration and then considers the ratio of blocking cost to duration. Another critical property, particularly in light of the approximate thresholds discussed above, is an overall tendency to aim for the threshold rather than prioritize one case type over another.

These three sets of observations suggest a framework for threshold-based heuristics. Stochastic dynamic programs like the SDSSP are known to be constrained by the 'curse of dimensionality.' Without pursuing a full analysis of the complexity of SDSSP, it is clear that the problem complexity grows as the initial OR capacity and the number of case types increase. This complexity makes heuristic approaches necessary for solving realistically sized problems in which the scheduled OR capacity may be as large as 8 h and the case durations may exceed 4 h.

4 Threshold-based heuristics

The proposed heuristics for the single-day surgery scheduling problem have two components: (1) a threshold for each day and (2) a case type prioritization rule. The thresholds identify how much open time to preserve for future primary cases and, for each state, define a target number of hours to take off the RQ. The case type prioritization rule then indicates how to reach this target.

4.1 Threshold determination

The trends in the approximate threshold behavior motivate three simple methods for determining thresholds and a fourth method based on the optimal threshold algorithm from SDSSP1. Each of these methods is described below:

- *Greedy*: Set each threshold to 0 h.
- *Day-to-Day*: Set each threshold to the expected number of hours of cases arriving that day.
- *Cumulative*: Set each threshold to the expected cumulative remaining number of hours of cases.
- *Smart*: Use the optimal threshold algorithm from SDSSP1.

The final method requires some manipulation of the input data in order to apply the SDSSP1 algorithm. Specifically, the multiple case types must be combined into a single case type. The arrival distributions for primary cases are combined to form a single distribution for the hours of arrivals on each day. The deferral and blocking costs for each case type are combined to form weighted costs, with the weights determined by the expected number of arrivals of each case type. The following definitions create the input data for the threshold algorithm.

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

$$\tilde{T}_j = \sum_{k=1}^{K} d_k T_j^k$$

$$\tilde{h}_j = \frac{\sum_{k=1}^{K} h_j^k E\left[T_j^k\right]}{E\left[\tilde{T}_j\right]}, \quad j = 0, \ldots, N$$

$$\tilde{r}_j = \frac{\sum_{k=1}^{K} r_j^k E\left[T_j^k\right]}{E\left[\tilde{T}_j\right]}, \quad j = 1, \ldots, N$$

For more details on the specific algorithm used to find the optimal thresholds, see the proof of the threshold optimality claim in Herring (2010).

The first three methods, which are based on the observed behavior for specific problem scenarios, should work well for some problems and poorly for others. Specifically, the *Greedy* threshold rule should work well for problems with "greedy" cost ratios. Similarly, *the Day-to-Day* and *Cumulative* threshold rules should work well for problems with the "low" and "high" cost ratios, respectively. The final threshold method can adapt to the input data and should perform well in all scenarios.

4.2 Case type prioritization

Three different case type prioritization schemes are based on the insights gleaned from the optimal solution behavior. The last two schemes described below are for days $1, \ldots, N$ only. Because the deferral and blocking cost structure is different on the day of surgery (a single penalty, r_0, for OR hours left empty rather than multiple blocking costs, r_j^k), these ratio dependent prioritizations take advantage of the knapsack nature of the day 0 costs (as described in Sect. 2).

- *Duration*: Case types are prioritized by decreasing duration.
- *Ratios*: Case types are prioritized first by decreasing deferral cost-to-duration ratio, and second by decreasing blocking cost-to-duration ratio. Remaining ties are ordered by decreasing duration.
- *Threshold First*: Cases are prioritized as in the *Ratios* rule, but the prioritization is subjugated to first reaching the target threshold.

This final prioritization scheme is motivated by the observation that the optimal decisions often place a higher priority on reaching the threshold than favoring one case type over another. However, implementing this final rule requires a simple integer program (IP). In addition to input data on the state of the RQ $\left(W_j^k\right)$ and the case types (d_k), the following definitions are needed for the IP:

D = target number of hours to take from the RQ and place on the OR schedule
p_k = priority of case type k

The priority values amount to a permutation of the set of case types $\{1, \ldots, K\}$, where a lower priority score denotes a preferred case type. Assuming that the case durations

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

and the scheduled OR capacity are integers, the IP can be defined as follows:

$$\min \sum_{k=1}^{K} (C+1)^{p_k} x_j^k + (C+1)^{K+1} z$$

$$\text{s.t.} \sum_{k=1}^{K} d_k x_j^k + z = D$$

$$0 \le {x_j}^k \le W_j^k, \quad {x_j}^k \text{ integer}, \quad k = 1, \ldots, K$$

$$z \ge 0$$

The slack variable z is given sufficient weight in the objective function to ensure that the target is reached if at all possible. If there is more than one way to hit the target, the RQ variables are weighted according to their priorities to ensure that more preferred cases are selected first. Note that the combination of base and exponent for the objective function weights reflects a kind of base $(C+1)$ arithmetic that ensures that no combination of less-preferred cases will be chosen over a more preferred case type.

As mentioned above, the *Ratio* and *Threshold First* prioritization schemes rely on the knapsack structure of the day 0 costs for the day of surgery. The *Ratio* scheme uses a greedy knapsack heuristic, ordering the case types by decreasing value-to-weight ratio and greedily taking as many of each type as possible. Because the *Threshold First* method already relies on the availability of an IP solver, the day 0 knapsack problem is solved to optimality with this prioritization scheme.

4.3 Heuristic quality

Combining the threshold determination rules and the case type prioritization schemes yields twelve heuristics. The problem instances explored in Sect. 3 are used to test these heuristics, and their solutions are compared with the optimal solution values. Notice that the blocking weight parameter (λ) and the primary arrivals case type prevalence $\left(E\left[T_j^2\right] : E\left[T_j^1\right]\right)$, which have little influence on the optimal solutions, play similarly limited roles in the heuristics. For this reason, the heuristics are tested on the original test instances that have $\lambda = 1$ and $E\left[T_j^2\right] : E\left[T_j^1\right] = 1 : 1$. This leaves a set of 18 test problems that vary only in their deferral and blocking cost structure.

An effective way to test the quality of the various heuristics is to imbed them in a simulation environment. The evolution of an OR schedule with specified input data and a RQ decision-making policy can easily be simulated to compute the total cost associated with a RQ policy and a particular realization of the arrival random variables. Each of the heuristics is simulated for 10,000 replications for each of the selected test problems, and the costs associated with each problem are expressed as percent deviations from the expected value of the optimal solution. Table 3 shows the mean deviation across all replications for each of the heuristics applied to the selected test problems. The 95% confidence interval (CI) half-widths for each of the values shown in Table 3 are given in Table 6 in the Appendix. Note that for all negative entries in the table, zero (representing the expected value of the optimal solution) falls within the CI. Recall that the three simple thresholds (*Greedy*, *Day-to-Day*, and *Cumulative*) are

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

Table 3 Average percent deviation from the optimal cost for twelve threshold-based heuristics applied to test problems representing a range of deferral and blocking cost structures

$h_j^2 : h_j^1$	1:1									Mean deviation	
$r_j^1 : h_j^1$	1:1			3:1			5:1			Target	All
$r_j^2 : r_j^1$	1:1	2:1	3:1	1:1	2:1	3:1	1:1	2:1	3:1		
Greedy											
Duration	*1.3*	*5.2*	*7.9*	19.6	50.3	87.2	61.5	129.2	194.9	4.8	61.9
Ratios	*2.8*	*7.7*	*9.4*	19.8	51.0	88.0	59.9	128.2	194.8	*6.6*	62.4
Threshold First	*18.0*	*18.8*	*17.5*	24.6	53.2	88.6	62.4	126.9	191.3	*18.1*	66.8
Day-to-Day											
Duration	50.0	29.2	15.1	*4.4*	*7.3*	*15.5*	11.4	31.8	48.2	*9.1*	23.7
Ratios	60.1	36.2	21.2	*9.1*	*9.9*	*17.3*	14.7	33.4	47.4	*12.1*	27.7
Threshold First	47.4	27.2	13.2	*2.9*	*5.6*	*14.1*	10.0	30.6	46.9	*7.5*	22.0
Cumulative											
Duration	96.7	62.2	39.0	11.6	4.4	3.7	*2.9*	*6.3*	*8.8*	*6.0*	26.2
Ratios	110.6	73.5	47.6	20.1	11.0	9.2	*10.7*	*12.3*	*16.0*	*13.0*	34.6
Threshold First	94.6	60.3	37.5	10.4	3.2	2.5	*1.8*	*5.3*	*7.8*	*5.0*	24.8
Smart											
Duration	1.3	31.7	20.8	8.3	4.4	3.7	2.9	4.2	3.8	n/a	9.0
Ratios	2.8	31.5	23.0	16.2	11.0	10.5	10.7	11.5	11.8	n/a	14.3
Threshold First	18.0	36.4	20.0	7.2	3.2	2.2	1.8	2.6	2.3	n/a	10.4
$h_j^2 : h_j^1$	2:1									Mean deviation	
$r_j^1 : h_j^1$	1:1			3:1			5:1			Target	All
$r_j^2 : r_j^1$	1:1	2:1	3:1	1:1	2:1	3:1	1:1	2:1	3:1		
Greedy											
Duration	*−0.2*	*−1.1*	*5.2*	3.0	24.4	46.8	31.7	75.3	126.6	*1.3*	34.6
Ratios	*7.8*	*−1.1*	*5.2*	5.1	24.4	46.8	33.2	75.3	126.6	*4.0*	35.9
Threshold First	*25.3*	*17.1*	*18.5*	11.3	29.5	50.2	35.4	77.4	127.0	*20.3*	43.5
Day-to-Day											
Duration	81.7	48.6	35.5	*6.0*	*2.1*	*4.5*	4.5	13.0	28.6	*4.2*	24.9
Ratios	89.0	48.6	35.5	*8.4*	*2.1*	*4.5*	4.8	13.0	28.6	*5.0*	26.1
Threshold First	81.7	48.6	35.5	*5.9*	*2.1*	*4.5*	4.4	13.0	28.6	*4.2*	24.9
Cumulative											
Duration	155.1	98.5	74.1	24.6	10.2	4.3	*6.4*	*3.5*	*6.1*	*5.3*	42.5
Ratios	156.8	98.5	74.1	25.2	10.2	4.3	*6.9*	*3.5*	*6.1*	*5.5*	42.8
Threshold First	155.1	98.5	74.1	24.6	10.2	4.3	*6.4*	*3.5*	*6.1*	*5.3*	42.5

Journal: **291** Article No.: **0270** TYPESET DISK LE CP Disp.:**2011/8/23** Pages: **31** Layout: **Small-X**

Table 3 continued

$h_j^2 : h_j^1$	2:1									Mean deviation	
$r_j^1 : h_j^1$	1:1			3:1			5:1			Target	All
$r_j^2 : r_j^1$	1:1	2:1	3:1	1:1	2:1	3:1	1:1	2:1	3:1		
Smart											
Duration	−0.2	−1.1	37.3	9.6	9.1	4.3	5.9	3.5	3.2	n/a	8.0
Ratios	7.8	−1.1	37.3	12.1	9.1	4.3	6.4	3.5	3.2	n/a	9.2
Threshold First	25.3	17.1	40.0	9.7	9.1	4.3	6.0	3.5	3.2	n/a	13.1

Averages across problems are shown both for targeted instances (indicated by italics) and all instances

expected to perform better for some instances than others. These targeted instances are shown in italics in Table 3, and the average performance of the heuristics across problem instances is shown for both the targeted instances and for all instances.

Several observations can be made from these results. First, the scenario-specific thresholds perform well when they should. In most cases, the cost of the solutions is less than 10% away from the optimal cost regardless of how the case types are prioritized. As might be expected, the quality of these thresholds deteriorates rapidly for other cost structures. The *Smart* threshold adapts to the different cost structures, with deviations typically falling at or below 10% and only exceeding 20% on a few of the "greedy" problems (those with $r_j^1 : h_j^1 = 1 : 1$). The *Smart* threshold outperforms the other thresholds as a general-purpose heuristic. Furthermore, its flexibility suggests that it will be able to conform and adapt to an even broader range of cost structures than those tested here.

None of the three prioritization schemes significantly outperforms the others across the entire range of problems. The *Threshold First* prioritization scheme is worse than the other prioritization schemes when paired with the *Greedy* and *Smart* thresholds and applied to the "greedy" problems. A greedy approach is appropriate for these problems, as observed from the policy maps, and this discrepancy between the prioritization schemes confirms the intuition that case type preference matters more when larger numbers of RQ cases are being placed. For the other subsets of problems, the *Duration* and *Threshold First* prioritization schemes slightly outperform the *Ratios* rule. Note that for the given case durations (1 and 2 h) the (decreasing) *Duration* rule will always reach the target if it is possible. This confirms the earlier observation that reaching the threshold is often more important than a particular case type preference.

4.4 Impact of arrival timing

The results for SDSSP1 (see Table 1) suggest that the timing of the primary case arrivals also impacts the target thresholds. In order to study how this affects the performance of the proposed heuristics, a second set of test problems is needed. The full range of inputs used in Table 3 (fixed blocking weight parameter, equal numbers of each case type expected over the course of the problem, and varying cost ratios) is combined

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

Table 4 Average percent deviation from the optimal solution for three threshold-based heuristics applied to test problems representing a range of primary service line arrival patterns and cost structures

	Early, Early			Middle, Middle			Late, Late			Early, Late			Late, Early		
$r_j^1 : h_j^1$	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1
Heuristic															
Greedy + Duration	1.9	15.8	40.0	−1.1	24.4	75.3	2.3	21.4	77.0	1.8	17.6	60.1	0.4	15.0	61.5
Smart + Duration	1.9	20.1	12.0	−1.1	9.1	3.5	2.3	6.4	0.4	1.8	5.5	3.5	0.4	1.3	3.8
Smart + Threshold First	14.7	20.3	12.1	17.1	9.1	3.5	28.8	6.4	0.4	18.3	5.5	3.5	19.3	1.3	3.8

with four new arrival patterns. Each of the two case types is set to arrive either "Early" or "Late," and all possible combinations are tested. The arrival random variables are still assumed to follow Poisson distributions for each day, with the "Early" arrival rates set to (1, 0.5, 0.25, 0.25, 0) and the "Late" arrival rates set to (0.25, 0.25, 0.5, 1, 0). This new test set consists of 72 problem instances.

Rather than apply all twelve of the proposed heuristics to this new problem set, three representative heuristics are chosen: *Greedy + Duration*, *Smart + Duration*, and *Smart + Threshold First*. The *Greedy + Duration* heuristic is chosen because it is the most naïve of the twelve and serves as a sort of worst case choice. The other two are chosen because they perform well over the entire initial problem set and most closely represent the trends observed in the policy maps. Table 4 shows the average performance of these heuristics over 10,000 simulated replications on the new set of test problems, again shown as the percent deviation from the optimal solution value. The evenly spread out arrival scenario in Table 3 is referred to as "Middle" and is included in the table for comparison's sake. In the interest of space, the results are only shown for single values of the between-case-type deferral and blocking ratios. Specifically, these ratios in Table 4 are fixed as $h_j^2 : h_j^1 = r_j^2 : r_j^1 = 2 : 1$ (the case duration ratio). While the exact magnitude of the deviations differs for other deferral and blocking cost structures, the observations and patterns discussed here remain constant across other cost structures. The 95% confidence interval half-widths for the results in Table 4 are given in Table 7 in the Appendix.

Many of the insights from the initial set of test problems are still evident in this second set of tests. As expected, the *Smart* thresholds tend to outperform the *Greedy + Duration* heuristic in all arrival scenarios except the "greedy" problems where the *Greedy* threshold is expected to perform well. On average, the *Smart + Duration* combination still slightly outperforms the *Smart + Threshold First* combination. Most of difference between these two heuristics originates from the "greedy" problems, while their performance is nearly identical for the other cost ratios.

It is also clear from the results in Table 4 that the timing of the primary service line's arrivals does have some impact on the performance of the heuristics. The *Greedy + Duration* heuristic performs better overall when more of the cases arrive early, while the other two heuristics perform worst in the (Early, Early) scenario. The early arrivals leave fewer arrivals closer to the day of surgery, which reduces the time that must be preserved for remaining cases and makes greedier policies more

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

appropriate. As observed with the first set of test problems, the *Smart* threshold suffers in scenarios where greedier policies are appropriate and more importance is placed on the case type preference. On the other hand, the *Smart* threshold paired with either case type preference does well when lower thresholds are required. In these scenarios, the case type preference plays a less significant role because fewer hours of cases are taken. While these observations appear at first to implicate the case type preferences as the weak step of the proposed heuristics, it may in fact be the case that the threshold behavior itself becomes less precise as the optimal solutions become "greedier."

4.5 Impact of block release dates

From the perspective of the single-day scheduling problem, block release dates act as constraints on RQ decisions and should lead to lower quality solutions. However, the extent of the impact will depend upon the nature of the optimal decisions. In particular, forcing some decisions to zero will have less impact on problems with high target thresholds and will have more impact on problems with low target thresholds.

In order to study the impact of block release dates, the 90 test problems presented in Tables 3 and 4 are combined to form a single set of problem instances. The feasible block release dates for these problems range from 3 days before surgery (day 3) to the day of surgery (day 0). Setting the block release date to day 3 does not constrain the solutions in any way because day 3 is the first day on which a RQ decision needs to be made (the system is empty on day 4). Once a block release date has been chosen, a RQ policy is needed to make decisions after the block is released. Three different RQ policies are combined with the block release dates: (1) the optimal decisions from the fully solved SDPs, (2) the *Greedy* + *Duration* heuristic, and (3) the *Smart* + *Duration* heuristic. Each of these twelve combinations is simulated over 10,000 replications for each problem instance and the average results are shown in Table 5. The results are again given as percent deviations from the unconstrained optimal solutions. As in Table 4, the results are only shown for single values of the between-case-type deferral and blocking ratios $\left(h_j^2 : h_j^1 = r_j^2 : r_j^1 = 2 : 1\right)$. The exact magnitude of the deviations for other deferral and blocking cost structures differs some, but the observations and patterns discussed here remain constant across other cost structures. The 95% confidence interval half-widths for the means in Table 5 are given in the Table 8 in the Appendix.

When paired with the optimal decisions, the day 3 results are consistently within 3 percent of the optimal expected value (with the 95% CI half-widths always including zero), verifying the simulation's ability to accurately gauge the performance of a given policy. When paired with the heuristic RQ policies, the day 3 results are directly in line with the unconstrained results presented in Tables 3 and 4, confirming for these instances that setting the block release date to day 3 is equivalent to having no block release date.

As the block release date moves closer to the day of surgery, its impact on solution quality becomes evident. Note that when the block release date is set to day 0, the choice of RQ policy loses significance because all the policies become greedy on the day of

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

Table 5 Average percent deviation from the unconstrained optimal solution for three RQ policies paired with all possible block release dates

	Early, Early			Middle, Middle			Late, Late			Early, Late			Late, Early		
$r_j^1 : h_j^1$	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1
RQ policy															
Block release															
Optimal decisions															
Day 3	2.9	−0.3	−1.6	−1.8	1.4	2.4	−0.9	0.5	0.6	2.6	1.4	−1.0	−1.1	−2.1	−0.6
Day 2	42.1	5.4	−1.3	38.0	6.1	2.5	47.5	8.5	0.8	44.6	7.0	−0.6	44.3	4.3	−0.3
Day 1	93.1	19.9	3.3	81.2	9.8	2.6	100.4	10.9	0.8	92.5	12.6	−0.5	100.6	11.7	1.3
Day 0	143.0	35.3	7.1	115.0	12.8	4.2	124.9	11.1	0.8	123.3	16.7	0.2	141.5	17.3	3.9
Greedy + Duration															
Day 3	5.3	15.2	39.8	0.7	23.6	75.2	1.6	22.5	77.0	8.2	19.9	58.2	−0.5	14.9	59.8
Day 2	43.4	8.2	8.9	40.6	21.4	52.0	50.8	37.8	83.3	52.5	27.9	52.1	44.8	15.1	41.4
Day 1	94.5	21.4	8.1	83.2	16.0	25.6	103.6	36.0	54.7	97.8	28.3	31.5	101.5	23.4	31.3
Day 0	143.0	35.3	7.1	115.0	12.8	4.2	124.9	11.1	0.8	123.3	16.7	0.2	141.5	17.3	3.9
Smart + Duration															
Day 3	5.3	17.1	9.2	0.7	8.2	5.5	1.6	8.8	0.8	8.2	8.1	2.7	−0.5	−0.6	1.5
Day 2	43.4	25.7	8.9	40.6	9.3	5.5	50.8	11.1	0.8	52.5	18.4	2.7	44.8	5.9	1.5
Day 1	94.5	34.9	12.6	83.2	9.8	3.5	103.6	11.1	0.8	97.8	18.4	2.7	101.5	13.8	2.2
Day 0	143.0	35.3	7.1	115.0	12.8	4.2	124.9	11.1	0.8	123.3	16.7	0.2	141.5	17.3	3.9

surgery. For the problems where greedier solutions are appropriate $\left(r_j^1 : h_j^1 = 1 : 1\right)$, the solutions deteriorate quickly as the block release policy becomes more restrictive. For other blocking-to-deferral cost ratios, the added costs of the block release dates depend heavily on the RQ policy they are paired with. When the block release dates are paired with good RQ policies (the optimal decisions or *Smart + Duration*), the added costs increase slightly as the block release policy becomes more restrictive, but rarely exceed 20% deviation from the optimal. In contrast, when the block release dates are paired with a poor RQ policy (*Greedy + Duration*) the solutions at times improve as the block release policy becomes more restrictive. This is most noticeable when the blocking-to-deferral cost ratio is high $\left(r_j^1 : h_j^1 = 5 : 1\right)$, which is exactly the scenario where the *Greedy* threshold performs poorly. This ability of block release dates to mitigate the effect of poor RQ policies adds a level of theoretical justification to their use in practice, where the overall quality of RQ policies may be unknown and difficult to ascertain.

5 Conclusions

This paper demonstrates that the threshold behavior that was shown to be optimal for the dynamic single-day surgery scheduling problem with unit case durations can be successfully extended to yield high quality solutions to the scheduling problem with

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

multiple case durations. Showing that the optimal solutions to SDSSP remain constant for large numbers of cases on the request queue maintains the computational tractability of the SDP and allows a diverse range of test problems to be solved to optimality. A detailed analysis of the optimal policy maps for these test problems reveals that the solutions follow an approximate threshold behavior and gives insight into how good thresholds and case type prioritization rules can be combined to form threshold-based heuristics.

A set of threshold-based heuristics is proposed, and the computational results show that heuristics using the optimal threshold algorithm from SDSSP1 consistently outperform heuristics using other threshold rules. While none of the proposed case type prioritization schemes dominates the others, the results favor schemes that focus on hitting the target thresholds rather than enforcing strict case type preferences. The nature of the target thresholds varies with both the deferral and blocking cost structure and the timing of the primary case arrivals. The results indicate that the threshold-based heuristics perform best when the target thresholds are higher (indicating that fewer RQ cases will be scheduled) and deteriorate somewhat when "greedy" (lower) target thresholds are appropriate.

Block release dates are shown to be increasingly costly for problems with "greedy" target thresholds. However, in scenarios with higher target thresholds, imposing block release dates is not costly if good RQ decisions are made after the block is released. Block release dates can improve solution quality when paired with poor RQ decisions (that is, the resulting solutions are not as bad as they would be otherwise). This observation provides practical justification for the use of traditional block release dates in practice.

This study of the single-day surgery scheduling problem is the first to model the sequential nature of an OR manager's daily block release and RQ decisions and to investigate the dynamic interaction of these decisions with the block schedule and primary case arrivals. The proposed threshold-based decision rules suggest a new way for hospitals and OR managers to look at block release and RQ policies. Rather than setting a block release date after which all unused OR time is released (as is done in practice), our threshold-based rules would gradually release unused OR time over the course of several days leading up to the day of surgery. These model-produced thresholds have the ability to adapt to differences in priority levels, contributions to revenue, and demand arrival patterns between surgical specialties and would help provide clear and empirical justification for an OR manager's decisions during the often contentious development of single-day surgery schedules.

Our modeling framework emerged from our discussions with collaborators in the peri-operative services department at UMMC, who have validated the importance of blocking and deferral costs in RQ decision-making, reviewed these results, and endorsed the concept of threshold-based policies. In order to develop specific threshold policies for this facility, we plan to use data from the hospital's surgical scheduling system databases to estimate the distributions of surgical demand arrivals for their OR suite. Additionally, we will continue to work with the appropriate OR stakeholders to develop reliable methods for determining accurate deferral and blocking cost values. These values have the potential to depend on differences in patient acuity levels, specialized resource needs, and revenue contributions between surgical specialties, and as

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

such will involve a combination of expert elicitation and data analysis. Finally, continued research in this area will use data from UMMC to compare schedules developed using our threshold-based policies with actual historical schedules on more traditional metrics such as OR utilization and throughput.

Appendix

See Tables 6, 7, 8.

Table 6 95% confidence interval half-widths for results presented in Table 3

$h_j^2 : h_j^1$	1:1								
$r_j^1 : h_j^1$	1:1			3:1			5:1		
$r_j^2 : r_j^1$	1:1	2:1	3:1	1:1	2:1	3:1	1:1	2:1	3:1
Greedy									
Duration	3.6	3.2	2.9	2.7	3.1	3.9	3.1	4.5	6.1
Ratios	3.8	3.4	3.0	2.7	3.2	4.0	3.1	4.5	6.2
Threshold First	3.8	3.4	3.0	2.7	3.2	4.0	3.2	4.5	6.1
Day-to-Day									
Duration	4.0	3.4	2.9	2.5	2.5	2.8	2.5	3.1	3.9
Ratios	4.5	3.7	3.2	2.7	2.7	2.9	2.7	3.2	3.9
Threshold First	4.0	3.4	2.9	2.5	2.5	2.8	2.5	3.1	3.9
Cumulative									
Duration	4.6	3.8	3.3	2.7	2.5	2.5	2.4	2.6	2.8
Ratios	5.1	4.3	3.6	3.0	2.7	2.7	2.7	2.8	3.1
Threshold First	4.6	3.8	3.2	2.6	2.5	2.4	2.4	2.6	2.8
Smart									
Duration	3.6	3.6	3.2	2.6	2.5	2.4	2.4	2.5	2.5
Ratios	3.8	3.7	3.3	2.9	2.7	2.7	2.7	2.8	2.8
Threshold First	3.8	3.7	3.2	2.6	2.5	2.4	2.4	2.4	2.5
$h_j^2 : h_j^1$	2:1								
$r_j^1 : h_j^1$	1:1			3:1			5:1		
$r_j^2 : r_j^1$	1:1	2:1	3:1	1:1	2:1	3:1	1:1	2:1	3:1
Greedy									
Duration	3.5	2.9	2.9	2.3	2.5	3.0	2.5	3.4	4.7
Ratios	3.7	2.9	2.9	2.4	2.5	3.0	2.6	3.4	4.7
Threshold First	3.8	3.1	3.0	2.4	2.6	3.1	2.6	3.5	4.7

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

Table 6 continued

$h_j^2 : h_j^1$	2:1								
$r_j^1 : h_j^1$	1:1			3:1			5:1		
$r_j^2 : r_j^1$	1:1	2:1	3:1	1:1	2:1	3:1	1:1	2:1	3:1
Day-to-Day									
Duration	4.4	3.5	3.2	2.3	2.2	2.3	2.2	2.5	3.1
Ratios	4.6	3.5	3.2	2.4	2.2	2.3	2.2	2.5	3.1
Threshold First	4.4	3.5	3.2	2.3	2.2	2.3	2.2	2.5	3.1
Cumulative									
Duration	5.5	4.3	3.8	2.7	2.4	2.3	2.3	2.3	2.5
Ratios	5.5	4.3	3.8	2.8	2.4	2.3	2.3	2.3	2.5
Threshold First	5.5	4.3	3.8	2.7	2.4	2.3	2.3	2.3	2.5
Smart									
Duration	3.5	2.9	3.3	2.5	2.3	2.3	2.2	2.3	2.3
Ratios	3.7	2.9	3.3	2.6	2.3	2.3	2.3	2.3	2.3
Threshold First	3.8	3.1	3.3	2.5	2.3	2.3	2.2	2.3	2.3

Table 7 95% confidence interval half-widths for results presented in Table 4

	Early, Early			Middle, Middle			Late, Late			Early, Late			Late, Early		
$r_j^1 : h_j^1$	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1
Heuristic															
Greedy + Duration	4.8	3.8	4.3	2.9	2.5	3.4	2.2	1.8	2.5	2.9	2.4	3.0	3.5	2.8	3.6
Smart + Duration	4.8	3.9	3.7	3.5	2.3	2.3	2.2	1.7	1.6	2.9	2.1	2.1	3.5	2.6	2.6
Smart + Threshold First	5.0	3.9	3.7	3.8	2.3	2.3	2.4	1.7	1.6	3.1	2.1	2.1	3.7	2.6	2.6

Table 8 95% confidence interval half-widths for results presented in Table 5

	Early, Early			Middle, Middle			Late, Late			Early, Late			Late, Early		
$r_j^1 : h_j^1$	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1
RQ policy															
Block release															
Optimal decisions															
Day 3	4.6	3.4	3.2	2.9	2.1	2.2	2.2	1.6	1.6	3.0	2.1	2.0	3.5	2.4	2.6
Day 2	5.1	3.5	3.2	3.3	2.3	2.2	2.6	1.8	1.6	3.3	2.2	2.0	4.1	2.6	2.6
Day 1	6.1	3.8	3.3	4.0	2.4	2.2	3.3	1.8	1.6	3.9	2.2	2.0	5.1	2.8	2.6
Day 0	7.3	4.1	3.3	4.7	2.4	2.3	3.7	1.8	1.6	4.3	2.3	2.0	6.1	2.9	2.6

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

Table 8 continued

	Early, Early			Middle, Middle			Late, Late			Early, Late			Late, Early		
$r_j^1 : h_j^1$	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1	1:1	3:1	5:1
Greedy + Duration															
Day 3	4.8	3.8	4.3	3.1	2.5	3.4	2.3	1.9	2.5	3.2	2.4	3.0	3.5	2.7	3.6
Day 2	5.2	3.7	3.7	3.5	2.6	3.3	2.7	2.1	2.7	3.5	2.6	3.1	4.1	2.8	3.6
Day 1	6.2	3.9	3.6	4.1	2.6	3.0	3.4	2.3	2.7	4.0	2.6	2.9	5.1	3.1	3.5
Day 0	7.3	4.1	3.3	4.7	2.4	2.3	3.7	1.8	1.6	4.3	2.3	2.0	6.1	2.9	2.6
Smart + Duration															
Day 3	4.8	3.9	3.6	3.1	2.3	2.4	2.3	1.8	1.6	3.2	2.1	2.1	3.5	2.5	2.6
Day 2	5.2	4.1	3.6	3.5	2.4	2.4	2.7	1.8	1.6	3.5	2.3	2.1	4.1	2.6	2.6
Day 1	6.2	4.3	3.7	4.1	2.4	2.3	3.4	1.8	1.6	4.0	2.3	2.1	5.1	2.8	2.6
Day 0	7.3	4.1	3.3	4.7	2.4	2.3	3.7	1.8	1.6	4.3	2.3	2.0	6.1	2.9	2.6

Proofs for Claims 1 and 2

See Sect. 2 for the definition of all variables, parameters, and functions used below.

Claim 1 *If, for a given state,* $W_j^k \geq \left|\frac{C_j}{d_k}\right|$, *then* $W_{j-1}^k \geq \left|\frac{C_{j-1}}{d_k}\right|$ *for all possible future states.*

Proof:

$$
\begin{aligned}
W_{j-1}^k - \left|\frac{C_{j-1}}{d_k}\right| &= \left(W_j^k - x_j^k + R_j^k + \left(T_j^k - fit_j^k\right)\right) - \left\lfloor \frac{C_j - \sum_{i=1}^K d_i \left(x_j^i + fit_j^i\right)}{d_k} \right\rfloor \\
&= W_j^k + \left(R_j^k + T_j^k\right) - \left(x_j^k + fit_j^k\right) \\
&\quad - \left\lfloor \frac{C_j - \sum_{\substack{i=1,\\ i\neq k}}^K d_i \left(x_j^i + fit_j^i\right)}{d_k} - \left(x_j^k + fit_j^k\right) \right\rfloor \\
&= W_j^k + \left(R_j^k + T_j^k\right) - \left\lfloor \frac{C_j - \sum_{\substack{i=1,\\ i\neq k}}^K d_i \left(x_j^i + fit_j^i\right)}{d_k} \right\rfloor \\
&\geq W_j^k - \left\lfloor \frac{C_j}{d_k} \right\rfloor \\
&\geq 0
\end{aligned}
$$

The first several steps are simply manipulations of the transition equations defined above. The second to last step uses the fact that the arrival random variables are

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

non-negative, as well as an observation on the nature of the floor function. The final step reflects the initial assumption that the desired inequality holds for day j. □

Claim 2 *For all feasible states* $\left(C_j, B_j, \boldsymbol{W}_j\right)$ *for all days* $j = 0, \ldots, N$, *the following equalities hold:*

(i) $F_j\left(C_j, B_j, \boldsymbol{W}_j, \boldsymbol{x}_j\right) = F_j\left(C_j, B_j, \widehat{\boldsymbol{W}}_{\boldsymbol{j}}, \boldsymbol{x}_{\boldsymbol{j}}\right)$
(ii) $V_j\left(C_j, B_j, \boldsymbol{W}_j\right) = V_j\left(C_j, B_j, \widehat{\boldsymbol{W}}_{\boldsymbol{j}}\right)$
(iii) $\boldsymbol{x}_j\left(C_j, B_j, \boldsymbol{W}_j\right) = \boldsymbol{x}_j\left(C_j, B_j, \widehat{\boldsymbol{W}}_{\boldsymbol{j}}\right)$

Proof: The proof proceeds by induction on j.

Base Case ($j = 0$): The first equality relies directly on the definition of $\widehat{W}_0^k$.

$$
\begin{aligned}
F_0\left(C_0, B_0, \boldsymbol{W}_0, \boldsymbol{x}_0\right) &= \sum_{k=1}^{K} h_0^k \left(\min\left(\left\lfloor \frac{C_0}{d_k} \right\rfloor, W_0^k\right) - x_0^k\right) \\
&\quad + r_0\left(C_0 - \sum_{k=1}^{K} d_k x_0^k\right) \\
&= \sum_{k=1}^{K} h_0^k \left(\widehat{W}_0^k - x_0^k\right) + r_0\left(C_0 - \sum_{k=1}^{K} d_k x_0^k\right) \\
&= F_0\left(C_0, B_0, \widehat{\boldsymbol{W}}_0, \boldsymbol{x}_0\right)
\end{aligned}
$$

Note that the feasible regions for the decision vector are identical for states $\left(C_0, B_0, \boldsymbol{W}_0\right)$ and $\left(C_0, B_0, \widehat{\boldsymbol{W}}_0\right)$. Both sides of equalities (ii) and (iii) represent minimization problems of identical functions over identical feasible regions, clearly implying that these equalities must hold.

Inductive Step: Assume that all the desired equalities hold for day $j-1$. Note that moving from state $\left(C_j, B_j, \boldsymbol{W}_j\right)$ to $\left(C_j, B_j, \widehat{\boldsymbol{W}}_{\boldsymbol{j}}\right)$ has no impact on the deferral and blocking penalties (ND_j^k, NB_j^k, and FB_j^k,) provided that the decision variables are held constant. The deferrals are computed based on what is feasible, the blocking penalties do not rely on the RQ outside of determining feasible decisions, and, as in the base case, the feasible regions are identical for both states. Therefore, looking at the difference between $F_j\left(C_j, B_j, \boldsymbol{W}_{\boldsymbol{j}}, \boldsymbol{x}_j\right)$ and $F_j\left(C_j, B_j, \widehat{\boldsymbol{W}}_{\boldsymbol{j}}, \boldsymbol{x}_j\right)$ comes down to looking at the difference between the subsequent day $j-1$ states. The next day's available space $\left(C_{j-1}\right)$ and blocking eligible hours $\left(B_{j-1}\right)$ will also be identical, again provided that the decision variables are held constant. Therefore, the focus lies on potential differences on day $j-1$ between the RQ states. The goal is to show that the trimmed day $j-1$ RQ states (limited to the day $j-1$ decisions that are feasible) are equal starting from either W_j^k or $\widehat{W}_j^k$ on day j. It will be helpful to express the RQ day j to day $j-1$ transition equation as a function of the starting RQ state, with all other variables being held constant.

$w\left(W_j^k\right)$ = subsequent day $j-1$ RQ state for case type k from state $\left(C_j, B_j, \boldsymbol{W}_{\boldsymbol{j}}\right)$

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

$w\left(\widehat{W}_j^k\right)$ = subsequent day $j-1$ RQ state for case type k from state $\left(C_j, B_j, \widehat{\boldsymbol{W}}_{\boldsymbol{j}}\right)$

Showing that these two RQ states have equal trimmed RQ states now amounts to showing that the following equality holds.

$$\min\left(\left\lfloor \frac{C_{j-1}}{d_k} \right\rfloor, w\left(W_j^k\right)\right) = \min\left(\left\lfloor \frac{C_{j-1}}{d_k} \right\rfloor, w\left(\widehat{W}_j^k\right)\right)$$

If $W_j^k = \widehat{W}_j^k$, then equality trivially holds. If $W_j^k \neq \widehat{W}_j^k$, then $W_j^k > \left\lfloor \frac{C_j}{d_k} \right\rfloor$ and $\widehat{W}_j^k = \left\lfloor \frac{C_j}{d_k} \right\rfloor$. Then by the results of Claim 1, $w\left(W_j^k\right) \geq \left\lfloor \frac{C_{j-1}}{d_k} \right\rfloor$ and $\left(\widehat{W}_j^k\right) \geq \left\lfloor \frac{C_{j-1}}{d_k} \right\rfloor$. As a result, both sides of the equality reduce to $\left\lfloor \frac{C_{j-1}}{d_k} \right\rfloor$.

Returning to equality (i), the difference in question becomes:

$$\begin{aligned}
&F_j\left(C_j, B_j, \boldsymbol{W_j}, \boldsymbol{x_j}\right) - F_j\left(C_j, B_j, \widehat{\boldsymbol{W}}_{\boldsymbol{j}}, \boldsymbol{x_j}\right) \\
&= E\left[V_{j-1}\left(C_{j-1}, B_{j-1}, \boldsymbol{w}\left(\boldsymbol{W}_j\right)\right) - V_{j-1}\left(C_{j-1}, B_{j-1}, \boldsymbol{w}\left(\widehat{\boldsymbol{W}}_{\boldsymbol{j}}\right)\right)\right] \\
&= E\left[V_{j-1}\left(C_{j-1}, B_{j-1}, \widehat{\boldsymbol{w}(\boldsymbol{W}_j)}\right) - V_{j-1}\left(C_{j-1}, B_{j-1}, \widehat{\boldsymbol{w}(\widehat{\boldsymbol{W}}_{\boldsymbol{j}})}\right)\right] \\
&= 0
\end{aligned}$$

The first step uses equality (ii) of the induction assumption, and second uses the result above that the trimmed day $j-1$ RQ states will be equal for both original day j RQ states. As in the base case, equalities (ii) and (iii) follow directly from equality (i). □

Pseudocode for *ProcessDay* algorithm (from Sect. 2)

INPUT: $C_j, B_j, \boldsymbol{x}_j, \boldsymbol{T}_j, \boldsymbol{d}$

DEFINE:

fit_j^k = number of primary arrivals of type k placed into the room on day j

$blfit_j^k$ = number of primary arrivals of type k that would have fit in the room on day j if the blocking eligible hours were available

INITIALIZE:

$\tilde{C}_j = C_j - \sum_{k=1}^K d_k x_j^k$ # space available in room after RQ decisions

$\tilde{B}_j = B_j + \sum_{k=1}^K d_k x_j^k$ # blocking eligible hours after RQ decisions

$\tilde{D}_j = \tilde{C}_j + \tilde{B}_j$ # space available if blocking eligible hours were free

FOR $k = K$ TO 1 # reflects prioritization of primary arrivals

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

determine cases of type k that either fit or would have fit into open space

$$tmpfit = \frac{\tilde{C}_j}{d_k}$$

$$fit_j^k = \min\left(T_j^k, \lfloor tmpfit \rfloor\right)$$

$$blfit_j^k = \min\left(T_j^k, \left\lfloor \frac{\tilde{D}_j}{d_k} \right\rfloor\right)$$

compute blocking penalties

$$NB_j^k = blfit_j^k - fit_j^k$$

$$FB_j^k = \max\left(0, blfit_j^k - tmpfit\right)$$

compute remaining space for next iteration

$$\tilde{C}_j = \tilde{C}_j - d_k fit_j^k$$

$$\tilde{D}_j = \tilde{D}_j - d_k blfit_j^k$$

END

RETURN: $\boldsymbol{fit}_j, \boldsymbol{NB}_j, \boldsymbol{FB}_j$

References

Beliën J, Demeulemeester E (2007) Building cyclic master surgery schedules with leveled resulting bed occupancy. Eur J Oper Res 176:1185–1204

Beliën J, Demeulemeester E, Cardoen B (2009) A decision support system for cyclic master surgery scheduling with multiple objectives. J Sched 12(2):147–161

Blake J, Carter M (2002) A goal programming approach to strategic resource allocation in acute care hospitals. Eur J Oper Res 140:541–561

Blake J, Donald J (2002) Mount Sinai Hospital uses integer programming to allocate operating room time. Interfaces 32(2):63–73

Brumelle S, Walczak D (2003) Dynamic airline revenue management with multiple semi–Markov demand. Oper Res 51:137–148

Cardoen B, Demeulemeester E, Beliën J (2009) Sequencing surgical cases in a day-care environment: an exact branch-and-price approach. Comput Oper Res 36(9):2660–2669

Cardoen B, Demeulemeester E, Beliën J (2010) Operating room planning and scheduling: a literature review. Eur J Oper Res 201:921–932

Denton B, Viapiano J, Vogl A (2007) Optimization of surgery sequencing and scheduling decisions under uncertainty. Health Care Manag Sci 10:13–24

Dexter F, Macario A, Traub RD (1999) Which algorithm for scheduling elective add-on cases maximizes operating room utilization? Use of bin packing algorithms and fuzzy constraints in operating room management. Anesthesiology 91:1491–1500

Dexter F, Traub R (2002) How to schedule elective surgical cases into specific operating rooms to maximize the efficiency of use of operating room time. Anesth Analg 94:933–942

Dexter F, Traub R, Macario A (2003) How to release allocated operating room time to increase efficiency: predicting which surgical service will have the most under-utilized operating room time. Anesth Analg 96:507–512

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X

913 Dexter F, Macario A (2004) When to release allocated operating room time to increase operating room
914 efficiency. Anesth Analg 98:758–762
915 Gerchak Y, Gupta D, Henig M (1996) Reservation planning for elective surgery under uncertain demand
916 for emergency surgery. Management Science. 42(3):321–334
917 Guinet A, Chaabane S (2003) Operating theatre planning. Int J Prod Econ 85:69–81
918 Gupta D (2007) Surgical suites' operations management. Prod Oper Manag 16(6):689–700
919 Hans E, Wullink G, van Houdenhoven M, Kazemier G (2008) Robust surgery loading. Eur J Oper Res
920 185:1038–1050
921 Herring WL (2010) A stochastic dynamic program for the single-day surgery scheduling problem.In: Pro-
922 ceedings of the 2010 IIE Society for Health Systems Annual Conference, Atlanta, 25–28 Feb 2010
923 Herring WL (2011) Prioritizing patients: stochastic dynamic programming for surgery scheduling and mass
924 casualty incident triage. Ph.D. Dissertation, University of Maryland, College Park, MD. Available at
925 http://hdl.handle.net/1903/11689
926 Lee TC, Hersh M (1993) A model for dynamic airline seat inventory control with multiple seat bookings.
927 Transp Sci 27(3):252–265
928 Macario A, Vitez T, Dunn B, McDonald T (1995) Where are the costs in perioperative care? Analysis of
929 hospital costs and charges for inpatient surgical care. Anesthesiology 83(6):1138–1144
930 McManus M, Long M, Cooper A, Mandell J, Berwick D, Pagano M, Litvak E (2003) Variability in surgical
931 caseload and access to intensive care services. Anesthesiology 98(6):1491–1496
932 Ozkarahan I (2000) Allocation of surgeries to operating rooms by goal programing. J Med Syst 24(6):339–
933 378
934 Pham D, Klinkert A (2008) Surgical case scheduling as a generalized job shop scheduling problem. Eur J
935 Oper Res 185:1011–1025
936 Santibañez P, Begen M, Atkins D (2007) Surgical block scheduling in a system of hospitals: an application
937 to resource and wait list management in a British Columbia health authority. Health Care Manag Sci
938 10:269–282
939 Schütz H-J, Kolisch R (2010a) Approximate dynamic programming for capacity allocation in the service
940 industry. Working paper, available at SSRN: http://ssrn.com/abstract=1618315
941 Schütz H-J, Kolisch R (2010b) Capacity allocation for demand of different customer–product–combina-
942 tions with cancellation, no–shows, and overbooking when there is a sequential delivery of service.
943 Working paper, available at SSRN: http://ssrn.com/abstract=1618313
944 Subramanian J, Stidham S Jr, Lautenbacher CJ (1999) Airline yield management with overbooking, can-
945 cellations, and no-shows. Transp Sci 33(2):147–167
946 Testi A, Tanfani E, Torre G (2007) A three-phase approach for operating theatre schedules. Health Care
947 Manag Sci 10:163–172
948 van Oostrum JM, van Houdenhoven M, Hurink JL, Hans EW, Wullink G, Kazemier G (2008) A mas-
949 ter surgical scheduling approach for cyclic scheduling in operating room departments. OR Spectr
950 30(2):355–374

Journal: 291 Article No.: 0270 TYPESET DISK LE CP Disp.:2011/8/23 Pages: 31 Layout: Small-X