

## Robust surgery loading

Erwin Hans<sup>a,\*</sup>, Gerhard Wullink<sup>b,1</sup>, Mark van Houdenhoven<sup>b</sup>, Geert Kazemier<sup>b</sup>

<sup>a</sup> *University of Twente, Department of Operational Methods for Production and Logistics, P.O. Box 217,  
7500 AE Enschede, The Netherlands*

<sup>b</sup> *Erasmus Medical Centre, Department of Operating Rooms, Intensive Care, and Anesthesiology, P.O. Box 2040,  
3000 CA Rotterdam, The Netherlands*

Available online 10 October 2006

---

### Abstract

We consider the robust surgery loading problem for a hospital's operating theatre department, which concerns assigning surgeries and sufficient planned slack to operating room days. The objective is to maximize capacity utilization and minimize the risk of overtime, and thus cancelled patients. This research was performed in collaboration with the Erasmus MC, a large academic hospital in the Netherlands, which has also provided historical data for the experiments. We propose various constructive heuristics and local search methods that use statistical information on surgery durations to exploit the portfolio effect, and thereby to minimize the required slack. We demonstrate that our approach frees a lot of operating room capacity, which may be used to perform additional surgeries. Furthermore, we show that by combining advanced optimization techniques with extensive historical statistical records on surgery durations can significantly improve the operating room department utilization.

© 2006 Elsevier B.V. All rights reserved.

*Keywords:* OR in health services; Scheduling; Heuristics; Operating theatre planning

---

### 1. Introduction

In the Netherlands, long waiting lists and ageing population will result in huge expenses for health care in the near future (TPG, 2004). Application of operations research techniques, which is common

practice in manufacturing industry, is expected to lead to an enormous improvement of efficiency in health care organizations. Erasmus MC in the Netherlands is one of the largest academic hospitals in Europe, with over 10,000 employees and an operating theatre with 16 operating rooms for inpatient surgeries. Every year approximately 9500 inpatient surgeries are performed, of which 85% are planned (elective). Since for a patient almost all costs occur on the day of surgery (Dexter, 2001), one of the most critical and expensive resources in a hospital is the operating theatre.

This paper addresses the problem of assigning elective surgeries to operating rooms (ORs), in such

---

\* Corresponding author. Tel.: +31 534893523; fax: +31 534892159.

*E-mail addresses:* [e.w.hans@utwente.nl](mailto:e.w.hans@utwente.nl) (E. Hans), [g.wullink@erasmusmc.nl](mailto:g.wullink@erasmusmc.nl) (G. Wullink), [m.vanhoudenhoven@erasmusmc.nl](mailto:m.vanhoudenhoven@erasmusmc.nl) (M. van Houdenhoven), [g.kazemier@erasmusmc.nl](mailto:g.kazemier@erasmusmc.nl) (G. Kazemier).

<sup>1</sup> Tel.: +31 104632528; fax: +31 104632098.

a way that not only the utilization of the OR theatre department is optimized, but also the total overtime is minimized. The latter prevents surgery cancellations. To make a surgery schedule more robust against overtime, we assign planned slack in addition to the planned surgeries on each OR-day. The planned slack is based on historical statistical data concerning the (stochastic) durations of the planned surgeries. We refer to this operational planning problem as robust surgery loading.

Erasmus MC has extensively collected data concerning the surgical process for more than 10 years. Various points in the process were consequently registered, which allows statistical analyses with respect to, e.g., surgery durations.

As is the procedure in many hospitals, in Erasmus MC each specialty has been assigned a number of ORs, to which they assign surgeries on a weekly basis. Unless there is a medical indication, these surgeries are assigned “FCFS,” using a “First Fit” priority rule (we shall discuss this method in Section 3.1). The surgery assignments are then verified by the OR-management, as to whether they are feasible with respect to capacity constraints, and as to whether the risk of overtime is not too high. To prevent overtime, the specialties must plan slack, which is based on the variability of the surgery durations of the specialty. The OR-management has decided that the amount of slack is such that the probability of overtime is approximately 30%. This amount is determined by assuming that the sum of the planned surgery durations is normally distributed: the amount of planned slack is then 0.5 times the standard deviation of the total planned surgery duration. This standard deviation is calculated from the given standard deviations of the individual planned surgery durations.

Several approaches for OR-planning have been proposed in the literature. Guinet and Chaabane (2003) first assign patients to ORs, and then reschedule the surgeries assigned to an OR to satisfy the material and personnel constraints. They use an extended version of the Hungarian method to solve an integer linear program that minimizes the patient waiting time between the hospitalization date and the actual start of the operation. Nevertheless, they do not consider the surgery duration variability.

Roth and Van Dierdonck (2005) propose an MRP-based approach that uses the concept of diagnostic-related groups (DRG) as a Bill of Materials. DRGs define health care as product classes, which comprise a relative standard set of activities and

required materials. In manufacturing planning and control literature it is generally recognized that MRP, which is material and not capacity oriented, has serious drawbacks, especially in environments characterized by much variability (see e.g. Hopp and Spearman, 2000; Zijm, 2000).

Beliën and Demeulemeester (2007) study the problem of building robust cyclic master surgery schedules, to minimize the shortage of beds. Demand constraints ensure that each surgeon obtains a specific number of ORs, and capacity constraints limit the number of OR blocks on each day. They formulate a stochastic MIP (with stochastic surgery durations), which they linearize and solve heuristically. De Kreuk et al. (2004) study the problem of developing a cyclic schedule for the activities of medical specialists. They use a composite objective function to optimize a schedule from the viewpoint of the specialist, by considering the location, preferred day-part, and the preferred sequence for each specialist. The resulting quadratic integer program (QIP) is solved using simulated annealing.

Kuo et al. (2003) deal with the surgery loading problem from the specialty’s viewpoint, and propose an LP model to optimize the revenues of the specialties. They assume that there are always sufficient cases to plan, intensive care capacity is unlimited, and surgeons are eager to operate more if this is possible. They do not consider surgery duration variability.

Dexter et al. (1999) study the problem of assigning add-on surgeries to ORs. Surgeries are planned based on their expected duration; no slack is planned to decrease the risk overtime. They compare 10 algorithms on the OR utilization for the on-line and off-line case. In the on-line case the surgeries are not known in advance, whereas in the off-line case they are. The algorithms are based on either “Best Fit” or “Worst Fit,” optionally with first sorting the surgeries on their duration (increasing or decreasing), and optionally with fuzzy constraints. Best (Worst) Fit assigns the surgery to the OR in which the available capacity is sufficient, but as small (large) as possible. The fuzzy constraints allow surgeries for which there is no OR in which they fit to be assigned to an OR in which at most 15 minutes of overtime is generated. Dexter et al. (1999) demonstrate that Best Fit Descending with fuzzy constraints performs best with respect to OR utilization, and they plead for using an off-line approach whenever possible. Goldman et al. (1969) demonstrated as early as 1969 that

“longest cases first” scheduling priorities yields the highest OR utilization rate, the lowest amount of overtime, and the largest number of delayed cases being transferred to another room.

Summarizing our short literature review on OR-planning, we conclude that there are several interesting approaches to planning problems in hospitals. The robust surgery loading problem has, in our opinion, not been addressed properly in the literature. While many approaches do recognize the presence of uncertainty in health care processes in general and OR-planning in particular, most do not deal with it explicitly. In this paper, we propose several constructive and local search heuristics for the robust surgery loading problem, which we regard as a “stochastic knapsack” problem. The objective is to optimize the assignment of surgeries by the specialties in such a way, that the risk of working in overtime is minimized, no surgeries are cancelled, and at the same time the OR capacity utilization can be improved.

The robust surgery loading problem is a generalization of the so-called “general bin packing problem with unequal bins,” which deals with assigning a set of items to a set of bins with different sizes. If necessary, the size of each bin can be extended. The objective is to minimize the sum of the sizes of the used bins. This problem (and hence also the surgery loading problem) is strongly NP-hard, which can be proven by reduction from three-partition. Dell’Olmo and Speranza (1999) prove that the longest processing time (LPT) heuristic has a worst-case bound of  $2 \cdot (2 - \sqrt{2}) \approx 1.17$  in the off-line variant of this problem, and a worst-case bound of  $5/4$  for list-scheduling (LS) in the on-line variant. Ye and Zhang (2003) propose an improved on-line algorithm, and discuss the problem in which the maximum item size may be larger than the smallest bin. They present a worst-case bound of  $6/5$ . The analogy with the surgery loading problem is that the ORs are bins with unequal capacities, which may be extended by planning overtime. The additional difficulty is that slack must be planned within the available capacity, which is based on the duration distributions of the concerned surgeries.

The remainder of this paper is organized as follows: Section 2 gives a formal problem description. Section 3 discusses the constructive heuristics, and Section 4 the local search heuristics. Section 5 gives the test results, and we conclude and outline directions for further research in Section 6.

## 2. Formal problem description

We consider a discretized planning horizon of  $T$  days ( $t = 1, \dots, T$ ), which in practice is usually a regular week of 5 days. On each day  $t$ ,  $K$  parallel identical operating rooms are available (index  $k = 1, \dots, K$ ). The capacity of operating room  $k$  on day  $t$  is  $c_{kt}$ . All operating rooms start at the same time. There are  $S$  specialties (index  $s = 1, \dots, S$ ). On each day  $t$ , each specialty  $s$  has a number of operating rooms at its disposal, indicated by the set  $K_{st}$  ( $\sum_s |K_{st}| = K, \forall t$ ). We refer to an element  $k$  of  $K_{st}$  as an “OR-day,” also denoted by a triplet  $(s, k, t)$ . For each OR-day an “OR-team” is available, which is a team of personnel required to perform surgeries. The personnel from whom an OR-team is put together comes from a so-called “unit.” A unit is an organizational unit, which contains personnel capable of supporting a number of specialties. We denote the set of specialties that can be supported by a unit  $u$  as  $O_u$  ( $u = 1, \dots, U$ ). We assume that each specialty has a sufficient number of surgeons on each OR-day to perform the surgeries. Hence on each day  $t$ , each specialty  $s$  has at least as many surgeons as it has OR-days at its disposal.

$N$  is the set of elective surgeries that are to be loaded ( $|N| = n$ ). Each surgery  $i$  ( $i = 1, \dots, n$ ) has an expected duration  $\mu_i$  (includes set-up time), and duration standard deviation  $\sigma_i$ .  $N_s$  is the set of surgeries performed by specialty  $s$ , and  $N = \bigcup_s N_s$ . In a given solution,  $N_{skt}$  ( $k \in K_{st}$ ) indicates the set of surgeries assigned by specialty  $s$  in operating room  $k$ , on day  $t$ . We do not consider emergency surgeries, or patients that must be planned on certain OR-days for a medical reason. These are planned *on-line*, while robust surgery loading is an *off-line* problem. Therefore, in our experiments, we have decreased the OR-day capacity  $c_{kt}$  to correct for these omitted emergency surgeries.

Without loss of generality, we assume that the elective surgeries  $N$  originate from an initial “base” solution, which specifies a set of assigned surgeries  $N_{skt}$  ( $k \in K_{st}$ ) for each OR-day  $(s, k, t)$ . The base solution serves as a reference point: it represents the OR program that Erasmus MC would use, if it would not have any of the techniques proposed in this paper at its disposal. The base solution is found using the First Fit dispatching rule, and some basic statistical data. In Section 3.1 we will explain this method, which is in fact the same as the one currently used by Erasmus MC for planning surgeries for regular (non-urgent) patients that are on the

waiting list. The reason that we use the surgeries from the base solution is that our loading methods will plan the same surgeries as the medical specialists would do in practice. In other words, no surgeries are cancelled, and no surgeries are added. By reloading and rescheduling surgeries, we are able to free capacity. The question whether this freed capacity should be used to plan additional surgeries, whether operating rooms should be closed earlier, or whether less personnel is required, is left to the board and the OR-manager.

The amount of planned slack on each OR-day to prevent overtime is based on the expected variance of the durations of the surgeries planned on that OR-day. The expected duration of the planned surgeries on OR-day  $(s, k, t)$ ,  $k \in K_{st}$  is

$$\mu_{skt} = \sum_{i \in N_{skt}} \mu_i \tag{1}$$

The variance of the planned surgeries on OR-day  $(s, k, t)$ ,  $k \in K_{st}$  is

$$\sigma_{skt}^2 = \sum_{i \in N_{skt}} \sigma_i^2 \tag{2}$$

We assume that the surgery durations are mutually independent. The planned slack size  $\delta_{skt}$  on each OR-day  $(s, k, t)$ ,  $k \in K_{st}$ , is calculated as follows:

$$\delta_{skt} = \beta \cdot \sqrt{\sum_{i \in N_{skt}} \sigma_i^2}, \tag{3}$$

in which  $\beta$  ( $\beta \geq 0$ ) is a parameter that influences the probability that the surgeries will complete on time, i.e., so overtime will not occur. In accordance with the central limit theorem in statistics (see e.g. [Kallenberg, 1997](#)) we assume that the sum of the durations of the planned surgeries on an OR-day is normally distributed with mean  $\mu_{skt}$  and standard deviation  $\sigma_{skt}$ . As a result, if for example  $\beta = 0.5$ , the surgeries will finish on time with a probability of 69.15%. Our approach can easily be adapted for other distributions, in which case the percentage will likely be different. In the literature often a log-normal distribution is chosen for the surgery duration ([Strum et al., 2000](#); [Zhou and Dexter, 1998](#)). However, the distribution of the sum of the surgery durations must then be approximated, since there is no known exact result for such a distribution. The value of  $\beta$  is typically chosen by management, since a higher  $\beta$  will lead to a lower OR utilization on the one hand, but may lead to less overtime, less costs,

and higher quality of labor and health care on the other hand.

Given a surgery allocation  $N_{skt}$ , the OR-day capacity constraint is as follows:

$$\sum_{i \in N_{skt}} \mu_i + \delta_{skt} \leq c_{kt} + O_{skt} \quad (\forall s, k \in K_{st}, t), \tag{4}$$

in which  $O_{skt}$  ( $O_{skt} \geq 0$ ) is the overtime on OR-day  $(s, k, t)$ . Observe we consider planned slack after regular time as overtime.

We distinguish between six degrees of freedom for the allocation of surgeries to OR-days, which we shall refer to as scenarios:

1. Surgeries must be planned on the day  $t$  that they were assigned to in the base solution, within the OR-days assigned to their specialty  $s$  ( $K_{st}$ ).
2. Surgeries must be planned on the day  $t$  that they were assigned to in the base solution, within the OR-days assigned to all the specialties supported by unit  $u$  ( $\bigcup_{s \in O_t} K_{st}$ ).
3. Surgeries must be planned on the day  $t$  that they were assigned to in the base solution ( $\bigcup_s K_{st}$ ).
4. Surgeries must be planned on the OR-days assigned to their specialty  $s$  ( $\bigcup_t K_{st}$ ).
5. Surgeries must be planned on the OR-days assigned to all the specialties supported by unit  $u$  ( $\bigcup_{t,s \in O_t} K_{st}$ ).
6. A surgery may be planned on any OR-day ( $\bigcup_{t,s} K_{st}$ ).

Given a scenario, the problem of assigning surgeries to operating rooms decomposes. For example in scenario 4, the problem decomposes into a sub-problem for each specialty. Scenarios 1–3 (as compared to 4–6) leave the surgery on the day it was in the base solution. Furthermore, the higher the scenario number, the more allocation freedom there is, and the more flexible the OR-personnel must be. More allocation freedom may give practical personnel and surgeon problems, as follows. In scenarios 1 and 4, OR-team and surgeon restrictions are automatically satisfied. In scenarios 2 and 5, OR-team restrictions are satisfied (since they can be exchanged freely between OR-days of a unit), but solutions may require more surgeons than available. In scenarios 3 and 6, both OR-team and surgeon restrictions may be violated. Since surgeries may be planned freely over all OR-days, this may result in solutions where more than  $K_{st}$  surgeries of specialty  $s$  are planned in parallel on day  $t$ . The sequence in which the surgeries are performed on

the OR-day is not of interest in this paper, since it does not affect the required OR-time. However, surgeon or OR-team restriction violations may be solved by changing the sequence in which the surgeries are performed, and/or by exchanging entire OR-day surgery assignments between different days. This is a subject for further research.

To evaluate a given solution, we use the following three ranked optimization criteria (in order of importance):

1. To minimize the total overtime:  $\sum_{s,k,t} O_{skt}$ .
2. To maximize the total number of free OR-days:  $\sum_{\{s,k,t|N_{skt}=0\}} 1$ .
3. To maximize the total free capacity:  $\sum_{s,k,t} \max\left(0, c_{kt} - \sum_{i \in N_{skt}} \mu_i - \delta_{skt}\right)$ .

If we compare two solutions, and both solutions score equally on criterion 1, then we use criterion 2 to compare, etc.

### 3. Constructive methods

#### 3.1. Base solution determination using First Fit

In this section, we describe the algorithm that we use to find a base solution. It is the loading procedure currently used by each specialty at Erasmus MC. There it is performed on Friday in week “ $x$ ,” and applies to 5 working days (Monday–Friday) of week “ $x + 2$ .”

Each specialty  $s$  allocates a number of surgeries ( $N_{skt}$ ) to each OR-day that has been assigned to them by management as follows. Each specialty has a waiting list of patients, which is sufficiently long to fill all available OR-days. The First Fit dispatching rule basically assigns the surgery from the top of the waiting list into the first OR in which it fits. A set of surgeries  $N_{skt}$  fits on an OR-day if the total expected surgery time plus the planned slack does not exceed the capacity  $c_{kt}$ . For the base solution (and currently in Erasmus MC), the planned slack is calculated differently than in Eq. (3). Instead of using the standard deviation  $\sigma_i$  of a surgery type  $i$ , each specialty  $s$  uses a standard deviation  $\sigma_s$  that is the same for all the surgeries performed by this specialty. As a result, the planned slack is computed as follows:

$$\beta \cdot \sqrt{\sum_{i \in N_{skt}} \sigma_s^2} = \beta \cdot \sqrt{|N_{skt}|} \cdot \sigma_s \quad (\forall s, k \in K_{st}, t),$$

in which  $\beta$ , just like in (3), is a parameter that determines the probability that the surgeries will complete on time, i.e., so overtime will not occur. The capacity constraint thus becomes

$$\sum_{i \in N_{skt}} \mu_i + \beta \cdot \sqrt{|N_{skt}|} \cdot \sigma_s \leq c_{kt} \quad (\forall s, k \in K_{st}, t). \quad (5)$$

The list of surgeries  $N_{skt}$  for an OR-day  $(s, k, t)$  is complete if no patients can be selected from the waiting list and added to  $N_{skt}$  without violating constraint (5), and enough slack can be planned during the OR-day. The algorithm requires  $O(n)$  time.

Our goal is to determine whether or not using more statistical information (i.e., the standard deviation  $\sigma_i$  per surgery type) and using more advanced planning techniques lead to an improved OR utilization.

#### 3.2. LPT-based dispatching

The longest processing time (LPT) dispatching rule is a list-scheduling (LS) variant in which the list of candidate items is first sorted in non-increasing order of size (for a comparison of LS and LPT, see Brehob et al., 2000). LS is commonly used for on-line parallel machine scheduling problems (we refer to the survey paper by Chen et al., 1998). LS is an on-line algorithm, because it permanently assigns the current job to a machine, before it is aware of the next job. Because it knows the job list in advance LPT is an off-line algorithm, which has a worst-case ratio of  $4/3 - 1/3m$  (Graham, 1969), where  $m$  is the number of machines. Without sorting the jobs, the worst-case ratio is  $2 - 1/m$  (Graham, 1966). “First Fit Decreasing” is an LPT-variant which assigns the item to the machine with the most time assigned so far, and the first one that fits.

We describe the LPT-based algorithm for scenario 4. The procedure is the same for the other scenarios, however with different ORs and surgeries. In scenario 4 the surgeries must be planned on the OR-days assigned to their specialty  $s$ , i.e., on all OR-days in:  $\bigcup_i K_{st}$ . This means we have to perform the algorithm precisely  $S$  times, since we have to solve an independent loading problem for each specialty  $s$ . As argued in Section 2, without loss of generality, we assume that the surgeries originate from the base solution (see Section 2). The list of surgeries is thus formed by the surgeries on the OR-days in  $\bigcup_i K_{st}$  in the base solution. We first sort these surgeries in non-increasing order of their expected duration. Hence, we essentially perform a longest expected

processing time (LEPT) rule. We then assign each surgery in this sequence to the first OR in which the surgery fits, i.e., for which constraint (4) still holds, without using overtime. If there is no OR in which the surgery fits without yielding overtime, it is assigned to the OR in which the additional overtime is as small as possible. The algorithm running time is  $O(n \log n)$ .

### 3.3. Sampling procedures

Sampling procedures have been successful as randomized constructive heuristics for resource constrained project scheduling problems Hartmann and Kolisch (2000). Just as list-scheduling, sampling procedures (generally) use a priority rule. The difference with LS is that sampling procedures use multiple passes. Different solutions are obtained by biasing the selection of the priority rule through a random device, and the best solution is kept after a number of passes. So, in addition to the surgery scheduling priority, a selection probability  $P_i$  is computed. Each pass is referred to as a sample. We consider three sampling methods: random sampling, biased-random sampling, and regret-based random sampling. The differences between these methods are only in the way the surgery drawing probabilities are calculated.

#### 3.3.1. General procedure in each sample

The procedure in each sample is as follows. The scenario determines what surgeries and OR-days are involved. We sort the list of surgeries in non-increasing order on their expected duration. Each iteration considers at most  $Z$  surgeries for dispatching from the beginning of the list ( $Z$  is a non-zero integer). If  $Z = 1$ , the algorithm is precisely the same as the LPT-based algorithm of Section 3.2, since only one surgery is considered in each iteration. For higher  $Z$ , the algorithm increasingly abandons the LPT-idea, and scours neighborhood solutions. The following procedure is carried out for each of the (at most)  $Z$  surgeries. If there are ORs in which surgery  $i$  fits without generating overtime, a scheduling priority  $v_{iskt}$  is calculated (we demonstrate how this is done later), and a most suitable OR is selected. If there are no ORs available in which surgery  $i$  can be planned without generating overtime, we immediately plan surgery  $i$  into the OR in which the generated overtime as in constraint (4) is as small as possible. We then replenish the list of  $Z$  surgeries. After a priority has been calculated

and a most suitable OR has been selected for each of the (at most)  $Z$  surgeries, a drawing probability  $P_i$  is calculated for each surgery  $i$ . Finally, a surgery is drawn and planned into the most suitable OR.

#### 3.3.2. Surgery scheduling priority calculation

We calculate a surgery scheduling priority  $v_{iskt}$  as follows. The idea is that we try to assign surgeries in such a way that the total planned slack is minimized. When a surgery is planned, it also introduces additional planned slack. If a surgery  $i$  is assigned to an *empty* OR, the additional planned slack is  $\beta \cdot \sigma_i$ , see Eq. (3). If a surgery  $i$  is assigned to a *filled* OR  $k$ , the additional planned slack is generally smaller. Suppose  $N_{skt}$  are the surgeries already assigned to OR  $k$ . Then the additional slack  $\Delta_{iskt}$  generated by adding surgery  $i$  is

$$\Delta_{iskt} = \beta \cdot \sqrt{\sum_{j \in N_{skt}} \sigma_j^2 + \sigma_i^2} - \beta \cdot \sqrt{\sum_{j \in N_{skt}} \sigma_j^2}.$$

The profit of not planning surgery  $i$  into an empty OR, but into a filled OR  $k$  is thus

$$\Omega_{iskt} = \beta \cdot \sigma_i - \Delta_{iskt}.$$

If  $(s, k, t)$  is a filled OR-day, then  $\Omega_{iskt} > 0$ . The priority of surgery  $i$  is

$$v_{iskt} = \max_k \Omega_{iskt}. \quad (6)$$

Of course Eq. (6) only involves the candidate OR-days, which depends on the scenario. The most suitable OR-day for surgery  $i$  is the  $k^*$  that maximizes  $\max_k \Omega_{iskt}$  in (6).

We minimize the total planned slack by exploiting the so-called portfolio effect. In the financial literature this term is used to indicate that portfolio risk falls with increasing diversity, as measured by the absence of correlation (covariance) between portfolio components (Markowitz, 1991). Since surgery durations are not correlated, we can minimize the total planned slack by clustering surgeries with similar variability on the same OR-day. To illustrate this, we give an example. Consider two OR-days, both with two assigned surgeries, one surgery with  $(\mu, \sigma) = (100, 10)$  and one surgery with  $(\mu, \sigma) = (100, 50)$ , see Fig. 1. We compare this situation (the left-hand side of Fig. 1) with the situation in which the surgeries with the same variability ( $\sigma$ ) are clustered. In the first situation, the standard deviation of the total duration of the surgeries is the same for both OR-days:  $\sqrt{(50^2 + 10^2)} = 51.0$ . The total planned slack is thus  $102.0 \cdot \beta$ . Similarly,

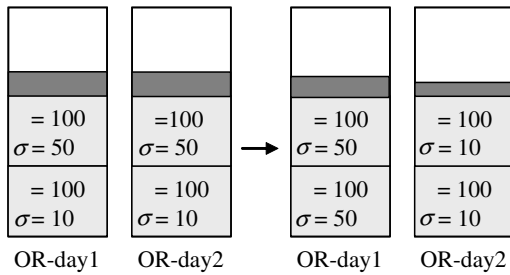


Fig. 1. Example of planned slack reduction as a result of the portfolio effect.

in the second situation the total planned slack is:  $(\sqrt{(50^2 + 50^2)} + \sqrt{(10^2 + 10^2)}) \cdot \beta = 84.9 \cdot \beta$ . This means a reduction in the total planned slack time of  $17.1 \cdot \beta$ , and thus an equal increase in the available capacity. This portfolio profit will increase when the variability of the concerned surgeries is higher.

We shall now describe the differences between the sampling methods.

### 3.3.3. Random sampling (S1)

S1 uses the aforementioned procedure to determine the most suitable OR, but discards the calculated surgery priority  $v_{iskt}$ . S1 gives each candidate surgery an equal probability to be loaded. Hence when there are  $q$  remaining candidate surgeries, each surgery  $i$  has a selection probability:  $P_i = 1/q$ . The drawn surgery is assigned to the most suitable OR.

### 3.3.4. Biased-random sampling (S2)

Biased-random sampling (Cooper, 1976) sorts the  $Z$  surgeries in order of non-increasing priority value. The  $i$ th surgery in the list then has a selection probability:  $P_i = C \cdot \gamma^i$ , in which  $C$  is a normalization constant ( $C = 1/\sum_j \gamma^j$ ), which ensures that the sum of surgery probabilities is 1, and in which  $\gamma$  ( $0 < \gamma \leq 1$ ) is a “bias factor.” Smaller values for  $\gamma$  result in a stronger dominance of the deterministic surgery priority values, and  $\gamma = 1$  corresponds to random sampling.

### 3.3.5. Regret-based random sampling (S3)

Biased-random sampling does not take into account the relative difference in priority values of activities in the list. To deal with this disadvantage, Drexel (1991) proposed regret-based random sampling (S3). In S3, the selection probability of a surgery is based on its “regret.” The regret  $w_{iskt}$  of a

surgery  $i$  is the difference between its priority  $v_{iskt}$ , and the worst of all surgery priorities:

$$w_{iskt} = v_{iskt} - \min_j \{v_{jst}\}.$$

The probability that a surgery  $i$  is selected is now

$$P_i = C \cdot (1 + w_{iskt})^\alpha, \quad (7)$$

in which  $C$  is a normalization constant that ensures that the sum of surgery probabilities is 1:

$$C = 1 / \sum_j (1 + w_{jst})^\alpha.$$

Furthermore,  $\alpha$  ( $\alpha \geq 0$ ) is the bias factor, i.e., a parameter that measures the bias.  $\alpha = 0$  corresponds to random sampling,  $\alpha = \infty$  corresponds to deterministic dispatching. Observe that in (7), the regret factor is incremented by 1, to ensure that all surgeries have non-zero selection probability. As a result, any surgery can be selected. Kolisch and Drexel (1996) show that deterministic sampling ( $\alpha = \infty$ ) works best when the sample size is 1. When the number of passes is increased, the sampling should be more biased, so  $\alpha$  must be decreased. We determine the best combination of  $\alpha$  and the number of samples empirically (see Section 5.3.1). Each sample of S1 and S3 requires  $O(n)$  time. Since S2 performs a sorting, it requires  $O(n \log n)$  time.

## 4. Local search methods

In addition to the constructive heuristics, we tested some local search heuristics, which focus on solution improvement by swapping two different surgeries between OR-days (a two-exchange), or by moving one surgery to another OR (a one-exchange). In Sections 4.1 and 4.2, we discuss two local search methods, which both use the aforementioned one- and two-exchanges.

### 4.1. Random exchange method

The random exchange method (REM) is a greedy local search procedure that uses the following procedure:

1. With probability  $P^{\text{REM}}$  a one-exchange is evaluated, for which we select a random surgery and a random suitable OR-day.
2. With probability  $1 - P^{\text{REM}}$  a two-exchange is evaluated, for which we select two random surgeries from two different OR-days.

If an exchange yields an improved solution, it is accepted. The method stops if no improvement has been found during  $\eta$  seconds.

#### 4.2. Simulated annealing

Simulated annealing (SA) is a generalization of a Monte Carlo method for examining the equations of state and frozen states of  $n$ -body systems (Metropolis et al., 1953). Kirkpatrick et al. (1983) propose it as an optimization technique for combinatorial problems. Ever since, SA has been widely and successfully used as a local search method in many applications. For an extensive description of the simulated annealing algorithm, we refer to Aarts and Korst (1989).

Our SA approach uses a proportional cooling scheme:  $\tau^{\text{new}} = \tau^{\text{old}} \cdot \theta$ , where  $\theta < 1$ , and  $\tau$  is the control parameter, which by analogy with the original application is also known as “cooling parameter,” or “temperature.” The initial control parameter is  $\tau^{\text{begin}}$ , and the final control parameter is  $\tau^{\text{end}}$ . The number of one- or two-exchanges evaluated before the control parameter is decreased (also known as the length of the Markov chain) is  $\lambda$ . We do a one-exchange with probability  $\pi$ , and a two-exchange with probability  $1 - \pi$ . Every exchange that yields an improvement of the solution (according to the three optimization criteria in Section 2) is accepted. If an exchange yields a worse solution, let  $Y$  be the increase of the solution value of optimization criterion 1, or, if this is 0, the increase of the solution value of optimization criterion 3 (we do not consider neighbor solutions in which the number of free days decreases). We accept the exchange

with probability:  $e^{-\frac{Y}{\tau}}$ . The cooling scheme parameters are such that the probability of accepting a worse solution is almost 1 at the start of the cooling scheme and is almost 0 at the end.

## 5. Test results

### 5.1. Test data

As argued in Section 2, we test our methods on an instance that is formed by a base solution, which we find using the “First Fit” based algorithm described in Section 3.1. We base the test instance on the operating theatre department of Erasmus MC, which consists of 16 ORs (for non-urgent clinical patients), and of 11 specialties. The instance spans a full year of 52 weeks, each week consisting of 5 working days of 7.5 hours. Hence the number of OR-days per year is  $52 \cdot 5 \cdot 16 = 4160$ , and the total surgery capacity is  $4160 \cdot 7.5 = 31,200$  hours. Table 1 gives an overview of the data used for each specialty.

Table 1 shows the OR allocation per surgery per week. For example on Monday, General surgery is assigned to OR 1, OR 2 and OR 3 (3 OR-days), Gynecological surgery to OR 4, etc. Table 1 also gives the standard deviation  $\sigma_s$  for all the surgeries performed by a specialty  $s$ , and the unit number to which the specialty belongs. For example Gynecological and Urological surgery are both supported by the OR-personnel of unit 4.

Since more than a decade the Erasmus MC has collected data concerning all processes in the organization, which is stored in a large data warehouse. This data warehouse contains medical information,

Table 1  
Instance data for the 11 specialties

Specialty	# OR-days per day					$\sigma_s$	Unit
	Monday	Tuesday	Wednesday	Thursday	Friday		
General surgery	3	3	3	3	3	105.8	2
Gynecological surgery	1	1	1	1	1	53.7	4
Oral surgery	1	1	1	1	1	71.2	3
ENT surgery	2	2	2	1	2	108.3	1
Pulmonary surgery	0	0	0	1	0	32.7	1
Neurosurgery	2	2	2	2	2	144.5	1
Traumatology	1	1	0	1	1	59.9	3
Eye surgery	1	1	1	1	1	31.8	1
Orthopedic surgery	1	1	2	1	2	73.5	3
Plastic surgery	2	2	2	2	1	110.8	3
Urological surgery	2	2	2	2	2	104.4	4
Total	16	16	16	16	16		

but also time-related data like the start and completion time of a surgery. Based on this data, we generate our artificial test instances. We generate an artificial instance because we do not consider the non-elective/emergency surgeries.

For each specialty, the surgeries are generated as follows. We made a statistical analysis of 10 years historical data of all elective inpatient surgeries in Erasmus MC. For each specialty, we clustered the surgeries into at least 4, and up to 8 surgery categories. Each surgery category has an expected surgery duration  $\mu_i$  and a duration standard deviation  $\sigma_i$ . The surgery category data is not given here, but is available from the authors upon request. To generate a surgery, a category is drawn with a probability that is proportional to the share of this category in all surgeries of the specialty. Consequently, the surgery characteristics are the same as the category characteristics. We use the procedure of Section 3.1 to generate a base solution, along with the aforementioned procedure to generate surgeries. If a generated surgery does not fit into the available OR-days, the OR-days are considered “filled.”

For the parameter  $\beta$  of constraint (3), which determines the probability that the surgeries will complete on time, we choose a value of 0.5. Assuming that the sum of the surgeries that are assigned to one day is normally distributed, this gives a probability of 69.15% that the surgeries will finish before the end of the planned slack. This is actually the probability that the board of Erasmus MC has chosen. The total number of surgeries in the test instance is 11,380. Of course, when  $\beta$  is decreased, more surgeries can be planned. For example, if  $\beta = 0$  (50% overtime probability), the number of surgeries is 13,470. On the other hand, if  $\beta = 3.9$  (0% overtime probability), the number of surgeries is 2158. Table 2 summarizes the characteristics of the base solution.

The total planned slack is computed using formula (3). Annual planned OR utilization rate (total surgery time plus slack/total capacity) is 83.20%.

Table 2  
Summary of the base solution characteristics

Number of surgeries	11,380
Total surgery time (hours)	23,177.45
Total OR capacity (hours)	31,200.00
Total overtime (hours)	0.0
Total unused/free capacity (hours)	5240.29
Total planned slack (hours)	2782.25

## 5.2. Test approach

All algorithms are implemented using the Borland Delphi programming language, and tested on a Pentium III-1600 desktop PC. We base all our experiments on the base instance described in Section 5.1. Depending on the scenario, an algorithm is applied to the OR-days and corresponding surgeries that belong to a specialty (scenarios 1 and 4), or a unit (scenarios 2 and 5), or all OR-days (scenarios 3 and 6), of a day (scenarios 1–3) or of a week (scenarios 4–6). So, for example in scenario 1, an algorithm must be applied precisely  $\#weeks \cdot \#days \cdot \#specialties = 52 \cdot 5 \cdot 11 = 2860$  times.

We first perform experiments in which we determine the best parameter settings for all the methods (Section 5.3). Given the best parameter settings for all methods, we then first compare the constructive methods (Section 5.4.1). We use the base solution and the best solution of all constructive methods and each scenario as starting solutions for the local search methods (Section 5.4.2). Finally, Section 5.5 gives the results of a Monte Carlo simulation of a solution found by the best algorithm.

## 5.3. Algorithm parameter determination

### 5.3.1. Sampling methods S1, S2, S3

For S1 we must set the parameter  $Z$ , the number of surgeries considered in each iteration, and the number of samples. We performed an experiment in which we test all values of  $Z \in \{1, \dots, 10, 20, 50\}$  in each scenario, and 10 samples. We compute the average objective values over all scenarios. We find the best average solution performance (over all three optimization criteria) is achieved for  $Z = 4$ . To determine the sample size, we make a trade-off between computation time and solution improvement. In an experiment in which we perform 1500 passes, we evaluate for each sample size in  $\{50, 250, 500, \dots, 1500\}$  the number of free days (optimization criterion 2), averaged over all scenarios (see Table 3). The total average overtime (criterion 1) is 0.0 in all samples. Table 3 shows that the solution improvements after 500 samples are marginal, so this is the value we choose for S1.

For S2, we must set the parameter  $Z$  and the bias factor  $\gamma$ . We performed an experiment in which we test all combinations  $Z \in \{1, \dots, 10, 20, 50\}$  and  $\gamma \in \{0.1, 0.2, \dots, 0.9\}$  in each scenario, and 10 samples. We compute the average objective values over

Table 3  
Computational results for S1

	Number of samples						
	50	250	500	750	1000	1250	1500
#Free days	336.0	341.3	343.2	344.7	345.0	345.5	345.8
Exec. time (seconds)	49.2	246.0	492.0	737.9	983.9	1229.9	1426.7

all scenarios. We find the best average solution performance for  $Z = 6$  and  $\gamma = 0.5$ . To determine the number of samples, we performed the same experiment as for S1. This gave very similar results, so we choose 500 samples for S2.

The parameters that we must set for S3 are:  $\alpha$  (bias factor), the number of samples, and  $Z$ . To determine these, we perform experiments with S3 in which we test all combinations of:  $\alpha \in \{1, \dots, 5, 10, 50, 100, 1000\}$ , 1500 samples, and  $Z \in \{1, \dots, 10, 20, 50\}$  for each scenario and the base instance. To determine the best values for  $\alpha$  and  $Z$ , we compare the average solution criteria values after 1500 samples for each  $(\alpha, Z)$  combination and in each scenario. The best solution performance is found for  $(\alpha, Z) = (10, 9)$ . Table 4 gives the results for S3 and the bias factor. Each cell is the average solution value (over all scenarios) for the optimization criterion of the row, and the bias factor of the column.

To determine the number of samples, we perform an experiment in which we vary the number of samples just as for S1 and S2, and with  $(\alpha, Z) = (10, 9)$ . We found that after 500 samples, the solution values only marginally improve, so this is the number of samples we choose for S3.

5.3.2. Random exchange method

For REM, we must choose the parameter  $P^{REM}$ , which determines the probability that an exchange is a one-exchange, and the stop-criterion, which is the number of seconds that no improvement was found ( $\eta$ ). The best solution performance was found for  $P^{REM} = 0.10$  and  $\eta = 2$ . To set  $\eta$ , we made a

trade-off between solution performance and computation time, both of which increase with  $\eta$ . For  $\eta > 2$ , the solution performance improvement is only marginal.

5.3.3. Simulated annealing

For SA, we determine the best values for the length of the Markov chain ( $\lambda$ ), the parameters of the proportional cooling scheme ( $\tau^{begin}, \tau^{end}, \theta$ ) and the fraction of one-exchanges ( $\pi$ ). For  $\lambda$ , we use the guideline suggested by Aarts and Korst (1989), the number of neighbor solutions. In our case, the number of neighbor solutions is

$$\lambda = \pi \cdot \varepsilon + (1 - \pi) \cdot \binom{\varepsilon}{2},$$

in which the first term is the number of one-exchanges, the second term is the number of two-exchanges, and  $\varepsilon$  is the number of surgeries considered for an exchange.

It is generally advised that the cooling schedule is such that the initial acceptance ratio (the number of accepted exchanges/the number of suggested exchanges) is close to 1 (see e.g., Van Laarhoven and Aarts, 1987). To determine the initial cooling parameter ( $\tau^{begin}$ ), we use the method suggested by Kirkpatrick et al. (1983). We choose an initial value 1 for  $\tau^{begin}$  and perform  $\lambda$  exchanges. If the acceptance ratio is below 0.8, we multiply  $\tau^{begin}$  by 2, and repeat the procedure until the acceptance ratio is above 0.8. The average best  $\tau^{begin}$  for the six scenarios we found is 256. Empirically, we also determined  $\pi = 20\%$ ,  $\tau^{end} = 0.001$ , and  $\theta = 0.995$ .

Table 4  
Average scenario objective values for S3 and various  $\alpha$

Optimization criteria (avg.)	Bias factor ( $\alpha$ )									
	0	1	2	3	4	5	10	50	100	1000
Overtime (hours)	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
# Free days	341.9	344.2	344.3	344.7	344.4	344.4	344.5	343.6	343.2	343.0
Free cap. (hours)	5370.8	5371.9	5371.9	5372.0	5372.0	5372.1	5372.4	5371.2	5371.7	5372.3

## 5.4. Algorithm comparison

### 5.4.1. Constructive methods

In Table 5 we compare the performance of the constructive methods, using the algorithm parameter settings discussed in Section 5.3. The last column gives the computation time for solving the entire base solution, averaged over all six scenarios. The other columns give the average solution performance for the three solution criteria.

Table 5 clearly shows that many OR-days can be freed by the constructive methods. This is especially achieved by the scenarios in which surgeries of different specialties are allowed to be combined into OR-days. Specialties that generally have long surgeries (such as neurosurgery) initially have a poor OR-utilization, because they lack short surgeries to fill their OR-days better. In scenarios in which surgeries of different specialties are allowed to be combined into OR-days, short surgeries from other specialties can complement these long surgeries, thus leading to free OR-days. S3 gives the best solution performance, albeit that its computation time is significantly higher. However, recall that the base solution spans a whole year. In a practical setting, the planning horizon is generally only a week. On average, S3 frees 6.7 OR-days every week, i.e., 8.4% of all OR-days.

From Table 6, in which we compare the objective values of S3 in each scenario, the profit of allowing more allocation freedom can be read. Compare, for

example, scenarios 1 and 4. The fact that we allow surgeries to be reassigned to OR-days of the specialty on another day makes that an additional 257 OR-days can be freed.

### 5.4.2. Local search methods

Table 7 compares the solution performance (avg. objective value over all scenarios) of the local search methods REM and SA, which start from the base solution.

SA clearly outperforms REM, and slightly outperforms S3, albeit with significantly more computation time. This might not be an issue in practice, if the OR-manager has sufficient time. We performed an additional experiment in which we executed REM after S3, to determine to what extent the S3 solution can be improved further. We found that on average only 3.2 additional OR-days and 14.2 hours of capacity can be freed. The combination S3+REM thus performs similar to SA, with much less computation time.

## 5.5. Simulation

In this section, we give the results of a Monte Carlo simulation of a solution found by the best algorithm. For each surgery, we draw the outcome of its duration, by sampling a value from its distribution function. Per OR-day, we then add up the surgery durations of all the planned surgeries, and compute the overtime or unused capacity. Table 8 gives the results of the Monte Carlo simulation of the base solution, and the solution for scenario 3 found by S3.

Table 5  
Average solution performance for the constructive algorithms over all scenarios

	Avg. overtime (hours)	Avg. #free days	Avg. free capacity (hours)	Avg. comp. time (seconds)
<i>Base solution</i>	0.0	0.0	5240.3	0.0
LPT	0.5	316.5	5359.7	0.0
S1	0.0	343.2	5373.4	492.0
S2	0.0	344.5	5376.5	632.8
S3	0.0	348.3	5372.9	2226.7

Table 6  
Objective values of S3 in each scenario

	Base sol.	Scen. 1	Scen. 2	Scen. 3	Scen. 4	Scen. 5	Scen. 6
Overtime (hours)	0.0	0.0	0.0	0.0	0.0	0.0	0.0
#Free days	0.0	5	135	552	262	508	628
Free capacity (hours)	5240.3	5258.9	5326.4	5445.3	5343.1	5409.9	5453.9

Table 7  
Performance comparison of the local search methods

	Avg. overtime (hours)	Avg. #free days	Avg. free capacity (hours)	Avg. comp. time (seconds)
<i>Base solution</i>	0.0	0.0	5240.3	0.0
REM (base sol.)	0.0	336.0	5380.1	1734.0
SA (base sol.)	0.0	352.5	5388.2	6315.6

Table 8  
Monte Carlo simulation results for scenario 3

	Total overtime (hours)	Avg. overtime per OR-day (hours)	OR overtime probability	Total free capacity (hours)	#Free OR-days	Avg. OR-utilization (%)
Base solution	330.1	0.08	0.094	8378.2	0	73.8
S3	886.0	0.21	0.255	8934.1	552	74.7

The third column in Table 8 gives the probability that any OR-day will result in overtime. The last column is the average OR-utilization of the *used* ORs, i.e. the average utilization of all OR-days with assigned surgeries. Observe that in the S3 solution, the probability that overtime occurs is higher, but still smaller than  $1 - 0.69 = 0.31$ , i.e. the selected probability that no overtime will occur (see Section 5.1). The reason that the actual probability is lower than  $1 - \beta$  is that there is still some unused capacity after the surgeries. We conclude that we can free 552 OR-days in scenario 3 (see Table 6 for the other scenarios) without exceeding the probability of overtime chosen by management.

## 6. Conclusions and further research

We have proposed constructive and local search heuristics for the robust surgery loading problem. The heuristics aim to minimize the total planned slack. Due to the portfolio effect, this automatically leads to freeing OR-days and -capacity. We have shown that a given plan that was made by specialists (the “base solution”) can be improved significantly by these methods.

The best constructive approach is regret-based random sampling (S3). The random exchange method (REM) can be used to further improve its solutions. Simulated annealing has a similar performance as S3, but uses much more computation time. All discussed approaches show that many OR-days and a lot of OR-capacity can be freed, without canceling surgeries, and without introducing more overtime than is allowed by the OR-management.

In the optimized solutions, we observed that as a result of the portfolio effect, surgeries with similar duration variability are often clustered on the same OR-day. Computational results show that smart algorithms and the exploitation of the portfolio effect frees much OR-capacity, within the limits of the accepted risk of overtime. This is only possible if extensive statistical data on surgery durations is available. This stresses the importance for hospitals

to register surgery durations (see also Ozkarahan, 2000), and OR-planners to recognize that surgeries have different duration variability, and account for this in OR-loading.

The clustering of surgeries with similar duration variability often leads to OR-days on which several surgeries of the same type are performed. This may have an additional advantage that surgeons can reduce surgery time because of the repetitive nature of their work on such OR-days. The clustering of surgeries can also be realized by using a so-called “master surgical scheduling” (MSS) approach (Blake and Donald, 2002; Beliën and Demeulemeester, in press), in which surgeries or surgery types are clustered in a (cyclic) schedule. In a forthcoming paper we shall present an MSS approach that accounts for both OR-capacity restrictions, and capacity restrictions imposed by subsequent departments, like ICU and wards (Van Oostrum et al., 2006).

In scenarios that allow much surgery allocation freedom, we ignored the surgeon and OR-personnel capacity restrictions. By optimizing the sequence in which the surgeries are performed on each OR-day, and by exchanging entire OR-day surgery assignments between days the extent to which these restrictions are violated can be decreased. This is also a subject for further research.

## References

- Aarts, E.H.L., Korst, J.H.M., 1989. Simulated Annealing and Boltzmann Machines. John Wiley & Sons, New York.
- Beliën, J., Demeulemeester, E., 2007. Building cyclic master surgery schedules with leveled resulting bed occupancy. *European Journal of Operational Research* 176 (2), 1185–1204.
- Blake, J., Donald, J., 2002. Mount Sinai hospital uses integer programming to allocate operating room time. *Interfaces* 32 (2), 63–73.
- Brehob, M., Torng, E., Uthaisombut, P., 2000. Applying extra-resource analysis to load balancing. *Journal of Scheduling* 3, 273–288.
- Chen, B., Potts, C.N., Woeginger, G.J., 1998. A review of machine scheduling: Complexity, algorithms and approximability. In: Du, D.Z., Pardalos, P.M. (Eds.), *Handbook of Combinatorial Optimization*, vol. 3. Kluwer Academic Publishers, pp. 21–169.

- Cooper, D.F., 1976. Heuristics for scheduling resource-constrained projects: An experimental investigation. *Management Science* 22 (11), 1186–1194.
- De Kreuk, A.C.C., Winands, E.M.M., Vissers, J.M.H., 2004. Master scheduling of medical specialists. In: Vissers, J.M.H., Beech, R. (Eds.), *Health Operations Management – Patient Flow Logistics in Health Care*. Routledge, London.
- Dell’Olmo, P., Speranza, M.G., 1999. Approximation algorithms for partitioning small items in unequal bins to minimize the total size. *Discrete Applied Mathematics* 94, 181–191.
- Dexter, F., 2001. Cost implications of various operating room scheduling strategies. *American Society of Anesthesiologist’s Clinical Update Program* 52 (262), 1–6.
- Dexter, F., Macario, A., Traub, R.D., 1999. Which algorithm for scheduling add-on elective cases maximizes operating room utilization?. *Anesthesiology* 91 1491–1500.
- Drexel, A., 1991. Scheduling of project networks by job assignment. *Management Science* 37, 1590–1602.
- Goldman, J., Knappenberger, H.A., Moore Jr., E.W., 1969. An application of OR scheduling policies. *Hospital Management* 107, 40–51.
- Graham, R.L., 1966. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal* 45, 1563–1581.
- Graham, R.L., 1969. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics* 17, 416–429.
- Guinet, A., Chaabane, S., 2003. Operating theatre planning. *International Journal of Production Economics* 85 (1), 69–81.
- Hartmann, S., Kolisch, R., 2000. Experimental evaluation of state-of-the-art heuristics for the resource-constrained scheduling problem. *European Journal of Operational Research* 127, 394–407.
- Hopp, W.J., Spearman, M.L., 2000. *Factory Physics – Foundations of Manufacturing Management*, second ed. McGraw-Hill/Irwin, New York.
- Kallenberg, O., 1997. *Foundations of Modern Probability*. Springer-Verlag, New York.
- Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., 1983. Optimization by simulated annealing. *Science* 220, 671–680.
- Kolisch, R., Drexel, A., 1996. Adaptive search for solving hard project scheduling problems. *Naval Research Logistics* 43, 23–40.
- Kuo, P.C., Schroeder, R.A., Mahaffey, M., Bollinger, R.R., 2003. Optimization of operating room allocation using linear programming techniques. *Journal of American College of Surgeons* 197 (6), 889–895.
- Markowitz, H.M., 1991. *Portfolio Selection: Efficient Diversification of Investments*, second ed. Blackwell, Cambridge, MA.
- Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, E., Teller, E., 1953. Equation of state calculations by fast computing machines. *Journal of Chemical Physics* 21 (6), 1087–1092.
- Ozkarahan, I., 2000. Allocation of surgeries to operating rooms by goal programming. *Journal of Medical Systems* 24 (6), 339–378.
- Roth, A.V., Van Dierdonck, R., 2005. Hospital resource planning. *Production and Operations Management* 4, 2–29.
- Strum, D.P., May, J.H., Vargas, L.G., 2000. Modeling the uncertainty of surgical procedure times: comparison of log-normal and normal models. *Anesthesiology* 92 (4), 1160–1167.
- TPG, 2004. Het kan echt: betere zorg voor minder geld. Eindrapportage TPG voor Sneller Beter – De logistiek in de zorg (in Dutch). Can be obtained from: <http://www.snellerbeter.nl>.
- Van Laarhoven, P.J.M., Aarts, E.H.L., 1987. *Simulated Annealing: Theory and Applications*. Reidel, Dordrecht.
- Van Oostrum, J.M., Van Houdenhoven, M., Hurink, J.L., Hans, E.W., Wullink, G., Kazemier, G., 2006. A master surgical scheduling approach for cyclic scheduling in operating room departments. Working paper, Erasmus MC, Rotterdam.
- Ye, D., Zhang, G., 2003. On-line extensible bin packing with unequal bin sizes. *Proceedings of the First Workshop on Approximation and Online Algorithms (WAOA 2003)*. Springer LNCS 2909, pp. 235–247.
- Zhou, J., Dexter, F., 1998. Method to assist in the scheduling of add-on surgical cases – upper prediction bounds for surgical case durations based on the log normal distribution. *Anesthesiology* 89, 1228–1232.
- Zijm, W.H.M., 2000. Towards Intelligent Manufacturing Planning and Control Systems. *OR Spectrum* 22, 313–345.