A variable neighborhood search algorithm for the surgery tactical planning problem

Citation for published version (APA):

Dellaert, N. P., & Jeunet, J. (2017). A variable neighborhood search algorithm for the surgery tactical planning problem. *Computers & Operations Research*, 84, 216–225. <https://doi.org/10.1016/j.cor.2016.05.013>

DOI:

[10.1016/j.cor.2016.05.013](https://doi.org/10.1016/j.cor.2016.05.013)

Document status and date:

Published: 01/08/2017

Document Version:

Typeset version in publisher's lay-out, without final page, issue and volume numbers

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

A variable neighborhood search algorithm for the surgery tactical planning problem

Nico Dellaert^a, Jully Jeunet^{b,*}

^a Technische Universiteit Eindhoven, School of Industrial Engineering, Eindhoven, The Netherlands

^b CNRS, Lamsade, université Paris Dauphine, place du Maréchal de Lattre de Tassigny, 75775 Paris Cedex 16, France

ARTICLE INFO

Article history:

Received 2 July 2015

Received in revised form

1 May 2016

Accepted 22 May 2016

Keywords:

Hospital capacity planning

Master surgery schedule problem

Multiple resources

Variable neighborhood search

ABSTRACT

We address the tactical planning problem of surgeries that consists in building an admission plan of patients over a medium-term horizon planning so as to minimize over and under utilization of several resources such as operating theaters, beds and nursing care, compared with their target level of utilization. The problem is formulated as a mixed integer linear program for which exact solution methods fail to find an optimal solution in a reasonable execution time. We develop a Variable Neighborhood Search algorithm and show its ability to provide high quality solutions in short computational running times compared with CPLEX for numerous real-sized instances based on the surgery planning problem in a Dutch cardiothoracic center. Furthermore, with few parameters' settings and low computational memory requirements, this approach may easily be implemented in a decision support system for hospitals.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

As in many organizations, decision making in hospitals occurs at three main levels: strategic, tactical and operational. Strategic decisions involve capacity planning decisions that allocate surgical specialities to operating-theater (OT) days for the long-term, often a year [1–3]. At the operational level, the problem is to schedule surgeries on a daily basis where patients are first assigned to days and surgery slots [4] or [5] and then sequenced for a given day [6] or [7,8].

In between the strategic and the operational levels lies the less tangible tactical level as hospital managers, submerged with real-time operational issues, are often inclined to solve problems at hand. They also claim for more capacity, a drastic strategic measure whereas a tactical allocation of available resources may be more effective and cheaper [9]. Such a tactical allocation problem is referred to the Master Surgery Schedule (MSS) problem and consists in allocating blocks of resource time (e.g. OT or nursing hours) to specialities and/or patient categories over a horizon length lying between the strategic and the operational horizon (e.g. a month).

A lot of tactical planning models focus on a single resource, the operating theater, as an increasing number of surgical procedures are now safely performed on an outpatient basis (recent surveys include [10–12]). However some surgeries like brain tumor

resection or cardiothoracic surgery procedures often imply a stay in the Intensive Care Unit (ICU) post surgery. For these surgeries, it is now widely recognized that integration of downstream resources such as beds or nursing care in ICUs leads to a better overall performance [3,13–15]. Thus, in addition to OT, some models to build MSSs include other resources such as nurses [16] or beds [17] or beds in the ICU and multiple general wards [18]. Adan et al. [14] consider cardiothoracic surgeries divided into several categories where the problem – which is under the scope of this paper – is to determine the number of patients in each category to be operated on for each day of the horizon. Four resources are included: OT, beds and nurses in the ICU and beds in the Medium Care Unit (MCU). This problem is formulated as a mixed integer linear program that can be solved to optimality with branch and bound algorithms. However, pitfalls of exact solving methods involve too long solution time. In Adan et al. [14] or Adan et al. [19] where emergencies were considered, several instances have been solved using CPLEX that failed to reach an optimal solution even after 24 h of running time. Obviously, taking into account multiple resources increases the problem complexity and a need for heuristic solution methods arises. As such, simulated annealing is used in Beliën et al. [17] and Fügner et al. [18]. Among local search techniques is the Variable Neighborhood Search (VNS) proposed by Mladenovic and Hansen in the late 1990s [20]. As this approach changes the neighborhood during the search, it is very successful compared to other local search techniques for solving many combinatorial problems from different areas [21]. Despite its effectiveness, the use of VNS to solve

* Corresponding author.

E-mail addresses: n.p.dellaert@tue.nl (N. Dellaert), jully.jeunet@dauphine.fr (J. Jeunet).

complex problems in health care delivery systems is limited to the nurse rostering problem [22,23].

In this paper we explore the potential of VNS to solve the tactical planning problem with multiple resources for cardiothoracic surgeries as formulated by Adan et al. [14] and also used in Adan et al. [19] and Dellaert et al. [24]. A solution to this problem is a tactical plan specifying for each day of a 28-day horizon (a month), the number of patients in each category to be operated on. Surgery of patients implies a consumption of the four resources for which capacity must not be exceeded. The objective function to minimize is a weighted sum of over and under utilization of the four resources compared with their target level of utilization. In our VNS, starting from an empty tactical plan, an initial solution is randomly generated using an insertion mechanism of patients on each day of the 4-week horizon. The neighborhood structure relies on the application of several local search operators on a feasible tactical plan, like swapping patients from different categories planned on two different days or shifting a single patient to another surgery day. The shaking procedure follows a ruin-reconstruct mechanism with deletion and subsequent addition of a variable number of patients in the current solution. Test results show that our algorithm outperforms CPLEX in most cases.

The remainder of this paper is organized as follows. Section 2 describes the tactical planning problem in detail. Section 3 presents the proposed VNS approach. Section 4 provides the computational study and discusses the results. Conclusions and suggestions for future research are given in Section 5.

2. Problem description

The problem is based on the functioning of the Thorax Center Rotterdam described in Adan et al. [14], where four critical resources are considered: operating theater hours (OT), number of beds in the ICU (IC), nursing hours in the ICU (NH) and number of beds in the MCU (MC). Patients are grouped in several categories that are homogeneous in terms of resource consumption. The patient flow in the Thorax Center is depicted in Fig. 1. Some categories of patients are admitted to the Medium Care Unit (MCU) one day before surgery. Each patient category is associated with a deterministic operation duration, based on average observed values. After surgery, patients are transferred to the ICU where they benefit from specialized nursing care. The nursing hours required for each patient category on each post-operative day are deterministic. After recovery, patients may stay in the MCU for a few days. Lengths of stay in the ICU and in the MCU are probabilistic, with different distributions for each category. Thus, a patient undergoing surgery on a given day consumes OT hours on that day, and has an expected consumption of bed and nursing hours in the ICU for a number of subsequent days corresponding to its length of stay in this unit. After the stay in the ICU, the patient has an expected consumption of bed in the MCU.

A feasible tactical plan thus specifies for each day of a 28-day horizon the number of patients in each category to be operated on, such that the consumption of each resource on any day does not

exceed the daily resource capacity. Furthermore, the total number of patients in each category planned for surgery over the whole horizon must be equal to the target throughput of patients which is set to the average number of surgeries observed in the past (over a same horizon length). The objective function is a weighted sum of over and under utilization of resources compared with their target level of utilization which is usually set to 80% of the capacity. The reason for using target deviations rather than cost relies on the fact that capacity allocation has been decided at the strategic level, so at the tactical level capacity is roughly fixed with a small variation margin though. Furthermore, target deviation is recognized as an important performance measure, besides waiting time and sojourn time in the hospital before treatment [25].

The mathematical formulation we use here was first given by Adan et al. [14] and then adapted to emergency cases in Adan et al. [19]. It was also used in Dellaert et al. [24]. We adopt the notation and definitions in Table 1.

Formally, the objective is to determine the values of variables $\{x_{c,t}\}$ satisfying a number of constraints and for which the daily expected utilization of each resource deviates as little as possible from the daily target. The objective function to be minimized can be written as

$$\sum_{r \in \{OT, IC, NH, MC\}} \alpha_r \sum_{t=1}^T (o_{r,t} + u_{r,t}), \quad (1)$$

Relative weights α_r assigned to the resources were derived from gross values of weights g_r as follows:

$$\alpha_r = \frac{g_r / \sum_{j=1}^T A_{r,j}}{\sum_{r \in \{OT, IC, MC, NH\}} (g_r / \sum_{j=1}^T A_{r,j})}, \quad (2)$$

with values for g_r decided by the stakeholders of the hospital, depending upon the degree of flexibility of each resource with the following rank: $g_{IC} > g_{OT} > g_{NH} > g_{MC}$.

Thus, finding an extra bed in the ICU was assessed as more difficult than calling for an additional surgeon. Beds in the MCU were considered as the most flexible resource since some patients can be discharged earlier than planned after surgery or may be sent to another hospital in order to free some beds. Nurses in the ICU were considered as a more flexible resource than surgeons, but less flexible than beds in the MCU. In our simulation experiment (Section 4), we make these weights varying.

For the sake of simplicity, we let $q_{r,t}$ be the expected consumption of resource r on day t . For all resources and periods, the expected consumption must not exceed the available capacity and over and under utilizations are auxiliary variables expressed as deviations between consumption and target level of utilization. Formally, the following constraints must hold

$$\begin{cases} q_{r,t} \leq K_{r,t} \\ -u_{r,t} \leq q_{r,t} - A_{r,t} \leq o_{r,t} \end{cases}, \quad r \in \{OT, IC, MC, NH\}; \quad t = 1, \dots, T. \quad (3)$$

The formulation of expected resources consumption is provided in Table 2 where we used the convention that subscript $t - j$

Fig. 1. The flow of elective patients in the Thorax Center.

Table 1

Notation.

Parameters	
N	Number of patient categories
c	Category index, $c = 1, \dots, N$
T	Length of the cyclic planning horizon, in days
t	Day index, $t = 1, \dots, T$
V_c	Target number of elective patients in category c to be operated on during the horizon
s_c	Surgery duration in hours for a patient in category c
l_c	Number of pre-operative days in the MCU for a patient in cat. c
r	Resource index, $r \in \{OT, IC, NH, MC\}$
$L_{ICU}^{\max}$	Maximum length of stay recorded in the ICU over all patient categories
$L_{MCU}^{\max}$	Maximum length of stay recorded in the MCU over all patient categories
$p_{ICU,c,j}$	Probability that a patient in category c is in the ICU, j days after surgery, $j = 0, 1, 2, \dots, L_{ICU}^{\max}$
$p_{MCU,c,j}$	Probability that a patient in category c is in the MCU, j days after surgery, $j = 0, 1, 2, \dots, L_{MCU}^{\max}$
$w_{c,j}$	Intensive care nursing workload (in hours) required for a patient in category c , j days after surgery
$K_{r,t}$	Maximum capacity for resource r on day t (expressed in number of hours for OT and NH and in number of beds for IC and MC)
$A_{r,t}$	Target utilization of resource r on day t
α_r	Relative importance of resource r as assessed by the stakeholders of the hospital
Variables	
$x_{c,t}$	Number of patients in category c planned for surgery on day t , with $c = 1, \dots, N$ and $t = 1, \dots, T$
$o_{r,t}$	Over utilization of resource r on day t , relative to its target level of utilization, with $r \in \{OT, IC, NH, MC\}$ and $t = 1, \dots, T$
$u_{r,t}$	Under utilization of resource r on day t , relative to its target level of utilization, with $r \in \{OT, IC, NH, MC\}$ and $t = 1, \dots, T$

in $x_{c,t-j}$ should be treated modulo T : day 0 is the same as day T , day -1 is the same as day $T-1$ and so on.

The total number of patients in group c to be operated on over the T -day cycle must be equal to the target patient throughput V_c . Values of V_c are obtained by rounding up the average number of patients operated on in each category based on historical data. Hence

$$\sum_{t=1}^T x_{c,t} = V_c, \quad c = 1, \dots, N. \quad (4)$$

As operating rooms on weekends are dedicated only to emergency patients, we have to require that

$$x_{c,t} = 0 \text{ and } x_{c,t+1} = 0, \quad t = 6 + 7(j-1); \quad j = 1, \dots, (T/7); \\ c = 1, \dots, N. \quad (5)$$

Our tactical planning problem therefore consists in minimizing the objective function in (1) subject to constraints (3), (4) and (5) with the following integrality constraints on the decision variables

Table 2

Formulation of expected consumptions.

Resource	Expected consumption
OT hours	$q_{OT,t} = \sum_{c=1}^N s_c \cdot x_{c,t}$
Beds in the ICU	$q_{ICU,t} = \sum_{c=1}^N \sum_{j=0}^{L_{ICU}^{\max}} p_{ICU,c,j} \cdot x_{c,t-j}$
Nursing hours in the ICU	$q_{NH,t} = \sum_{c=1}^N \sum_{j=0}^{L_{ICU}^{\max}} w_{c,j} \cdot p_{ICU,c,j} \cdot x_{c,t-j}$
Beds in the MCU	$q_{MC,t} = \sum_{c=1}^N \sum_{j=1}^{l_c} x_{c,t+j} + \sum_{c=1}^N \sum_{j=0}^{L_{MCU}^{\max}} p_{MCU,c,j} \cdot x_{c,t-j}$

$$x_{c,t} \in \{0, 1, 2, \dots, V_c\}, \quad c = 1, \dots, N; \quad t = 1, \dots, T. \quad (6)$$

The problem has therefore $N \cdot T$ integer decision variables and $12T + 9N$ constraints. In the worst case the number of nodes to explore in the branch-and-bound would equal $\sum_{c=1}^N (V_c + 1)^T$ which represents more than $2 \cdot 04^{51}$ nodes for the Thorax Center problem with 111 patients dispatched in 8 categories to be operated on a 28-day horizon.

3. The VNS for the surgery tactical planning problem

Our Variable Neighborhood Search for the tactical planning problem relies on the main steps of the classical general VNS described by Hansen and Mladenovic [21] but differs in some points. To emphasize the differences, both algorithms are provided in same Table 3 where R-VNS designates our VNS and FD-VNS refers to the classical approach with a first descent strategy. These differences are related to the application of the shaking procedure and the local search method. Neighborhood N_k is obtained through a shaking procedure that uses a ruin-reconstruct mechanism described in Section 3.1. Contrary to FD-VNS, in R-VNS this shaking procedure only applies when the local search did not improve the best solution so far. Neighborhood N_l corresponds to the application of the Π tH local search operator in a sequence of several operators defined in Section 3.2. In FD-VNS operators are explored in order and the process iterates until a local optimum is found at each neighborhood. Only strictly better solutions are accepted. In contrast in R-VNS the sequence of operators is repeatedly applied on a current solution and each operator is performed only once with random modifications that are accepted as long as they are feasible. Section 3.2 further develops the two local search methods. An initial solution is generated by randomly adding patients one by one to an empty plan, following the method explained in Section 3.1.

3.1. Initial solution and shaking

Initial solution: Starting from an empty tactical plan, we add patients one by one. We first choose a day t at random in such a way that days with higher under utilization of OT resource have higher chances to be selected. Then a category has to be chosen. The lower the target deviation of OT resource on day t after adding a patient in category c , the higher is the probability to select this category. Next, the feasibility of adding a patient in category c on day t is checked: consumption of all resources over the days corresponding to the sojourn of this additional patient must not exceed the available capacity. If feasible, the addition is implemented and the solution is updated, with an incremental computation of the objective value. If the adding operation is infeasible, probability of selecting category c is set to zero and probabilities for other categories are rescaled. Another category is then picked at random and the procedure is applied again until feasibility is possibly reached. If all categories are infeasible on day t , the solution is reset to zero and the procedure of adding patients is applied again.

Actually, simulations showed that finding a feasible initial solution was quite fast and rarely asks for more than one pass. It should be noted that besides this random initialization we also ran CPLEX for a limited execution time to get a good quality initial solution.

Shaking: Contrary to the classical general VNS, the shaking procedure in our VNS only applies when the local search did not improve the best solution so far. This means that when a new best solution is found, the local search is re-applied so as to intensify the search around x_{best} .

The shaking procedure follows a ruin-reconstruct mechanism. It consists in deleting a number of patients one by one from the

Table 3

Algorithms of the classical general VNS (FD-VNS) and the proposed VNS (R-VNS) for the tactical planning problem.

Classical general FD-VNS	Proposed R-VNS
Initialization <ul style="list-style-type: none"> Define N_k, $k = 1..k_{max}$ Define N_l, $l = 1..l_{max}$ Generate an initial solution x_0 Set $x_{best} = x_0$ While time limit is not exceeded $k = 1$ While $k \leq k_{max}$ Shaking We obtain $x \in N_k(x_{best})$ Local search with first descent Set $\begin{cases} x_{local} = x \\ l = 1 \end{cases}$ While $l \leq l_{max}$ Explore all possible modifications on x suggested by operator l . The modified solution is $x' \in N_l(x)$ Move or not If x' better than x_{local} set $\begin{cases} x_{local} = x' \\ l = 1 \end{cases}$ otherwise set $l = l + 1$ Move or not If x_{local} better than x_{best} set $\begin{cases} x_{best} = x_{local} \\ k = 1 \end{cases}$ otherwise set $k = k + 1$	Initialization <ul style="list-style-type: none"> N_k: delete and add k patients, $k = 1..k_{max}$ N_l: apply the l^{th} search operator, $l = 1..l_{max}$ Generate an initial solution x_0 Set $x_{best} = x_0$ While time limit is not exceeded $k = 0$ While $k \leq k_{max}$ Random local search Set $\begin{cases} x_{local} = x \\ m = 1 \end{cases}$ While $m \leq m_{max}$ Apply to x the l^{th} operator, $l = m - 1 \pmod{l_{max}} + 1$. We obtain $x' \in N_l(x)$ If x' feasible, set $x = x'$ Move or not If x better than x_{local} set $\begin{cases} x_{local} = x \\ m = 1 \end{cases}$ otherwise set $m = m + 1$ Move or not If x_{local} better than x_{best} set $\begin{cases} x_{best} = x_{local} \\ x = x_{best} \\ k = 0 \end{cases}$ otherwise Set $k = k + 1$ Shaking We obtain $x \in N_k(x_{best})$

incumbent solution in periods selected at random so that those with higher over utilization of OT resource have higher chances to be picked. Once a period is selected, a planned category in that period is randomly chosen according to the best OT deviation the deletion would lead to. We thus get a pool of removed patients that are then reinserted one by one in the solution, using the same adding procedure as the one we implemented in the initial solution procedure. We considered a maximum number of deletions k_{max} set to a percentage α of the total number of patients V , $k_{max} = \alpha V$.

3.2. Local search

In the classical general VNS, the local search adopts a first descent (FD) or best improvement strategy. All possible moves of a move type are explored until the first or the best improving move is found. If no improvement is reached, the next move type is explored. In our VNS, each local search operator is applied once and if feasible, the move is systematically implemented whether improving or not. The sequence of operators is applied several times to the current solution which is at least modified m_{max} times. The operators are described hereafter.

CategoryExchange: Two periods (t_a , t_b) exchange their planning of all patients in one category c_a , where days t_a and t_b have greater chances to be selected when over utilization on day t_a and under

utilization on day t_b of all resources are higher. Planned categories with higher number of patients on day t_a than that of day t_b are candidates and picked at random according to their target throughput: the more crowded a category is, the higher its probability to be selected is.

Swap: In the swap, one patient in category c_a planned on day t_a is moved to day t_b and one patient in category c_b planned on day t_b is moved to day t_a . Two versions of this swap based on different fitness measures are implemented. These measures relate to the impact of the move on the two most important resources, namely OT and IC.

SwapOT: Day t_a is randomly chosen according to its OT over utilization: the higher the OT over utilization is, the higher the probability of selecting the corresponding day is. We then determine the best combination of categories (c_a , c_b) to be swapped. This combination is such that deleting one patient in category c_a and adding one patient in category c_b on day t_a lead to the best OT target deviation on that day. Next we choose a day t_b with OT under utilization and at least one patient in category c_b : the lower the OT target deviation after adding one patient in c_a and deleting one patient in c_b on a given day is, the higher is the probability to select that day.

SwapIC: Two categories (c_a , c_b) with different surgery durations are selected uniformly at random. Random selection of day t_a (with at least one planned patient in category c_a) is based upon the best IC target deviation resulting from the move (deletion of one patient in c_a and addition of one patient in c_b) on the first subsequent period where the probabilities of occupying a bed in the ICU post surgery for categories c_a and c_b differ. Selection of day t_b is implemented in a similar way, considering the opposite move.

Shift: In the shift, one planned patient in category c_a on day t_a is shifted to day t_b . Again, two versions of this operator are considered.

ShiftOT: Like in SwapOT, day t_a is picked at random according to its OT over utilization. Selection of category c_a planned on day t_a is made in the following way: the lower the target deviation of OT after deleting one patient of a given category is, the higher the probability to select this category is. Next, the probability to choose day t_b is decreasing with the OT deviation after the addition of one patient in category c_a .

ShiftIC: To introduce more diversity, this shift uses similar selection mechanisms to those of CategoryExchange. Days t_a and t_b have greater chances to be selected that IC over utilization on day t_a and IC under utilization on day t_b are higher. Category c_a has higher probability of being picked that its target throughput is higher.

For any single patient move (ShiftOT and ShiftIC), feasibility is checked locally by comparing the updated consumption of resources implied by this additional patient with the available capacity, only over the days of this patient sojourn. Feasibility is checked globally for other moves (swaps and category exchange) as they imply a plan change for more than one patient.

We ranked the operators from the most disruptive to the least one and kept the same following order throughout in which CategoryExchange is called twice for the sake of diversity: {CategoryExchange, CategoryExchange, SwapOT, SwapIC, ShiftOT, ShiftIC}. We thus have $l_{max} = 6$. The solution was updated by taking into account only the changes so the objective function value did not need to be re-computed entirely. Updated consumptions were compared with target utilization levels to determine the values of over and under utilization for all resources so as to compute the objective function value.

Table 4 gives a mathematical description of the operators and the selection probabilities of days and categories. We also provide the deterministic version of these operators that will be used in FD-VNS for which we have $l_{max} = 5$ since there are 5 different operators.

The first column in Table 4 displays for each operator the elementary operations performed on the planning of patients. The second column indicates the selection type for categories and days, where D stands for deterministic and R for random. The next

Table 4

Selection of days and categories in the local search operators.

Operator	Sel.	First element	Second element	Third element
Cat. exch.				
$x_{ca,t_a} := x_{ca,t_b}$	D	$t_a \mid \sum_r o_{r,t_a} > 0$	$t_b \mid \sum_r u_{r,t_b} > 0$	c_a $c_a \mid x_{ca,t_a} > x_{ca,t_b}$
$x_{ca,t_b} := x_{ca,t_a}$	R	$p(t_a) = \frac{\sum_r o_{r,t_a}}{\sum_t \sum_r o_{r,t}}$	$p(t_b) = \frac{\sum_r u_{r,t_b}}{\sum_t \sum_r u_{r,t}}$	$p(c_a) = \frac{V_{ca}}{\sum_{c \mid x_{c,t_a} > x_{c,t_b}} V_c}$ and condition above
SwapOT				
$x_{ca,t_a} := -1$ $x_{ca,t_b} := +1$ $x_{cb,t_a} := +1$ $x_{cb,t_b} := -1$	D	$t_a \mid o_{OT,t_a} > 0$	$c_a \mid x_{ca,t_a} > 0$	$t_b \mid x_{cb,t_b} > 0$
	R	$p(t_a) = \frac{o_{OT,t_a}}{\sum_t o_{OT,t}}$	$c_a, c_b \mid d_{OT,t_a} \min$ $p(c_a), p(c_b), p(t_a)$ uniform and cond. above	$d_{OT,t_b} \min$
SwapIC				
Same as SwapOT	D	All	$c_b \neq c_b \mid s_{cb} = s_{ca}$	$t_a \mid x_{ca,t_a} > 0$
	R	$p(c_a) = 1/N$	$p(c_b)$ uniform	$t_b \neq t_a \mid x_{cb,t_b} > 0$ $p(t_i) = 1 - \frac{d_{IC,t_i+1}}{\sum_{t \mid x_{c,t} > 0} d_{IC,t+1}}$ $i = a, b$
ShiftOT				
$x_{ca,t_a} := -1$ $x_{ca,t_b} := +1$	D	All	$c_a \mid x_{ca,t_a} > 0$	$t_b \neq t_a$
	R	$p(t_a) = \frac{o_{OT,t_a}}{\sum_t o_{OT,t}}$	$p(c_a) = 1 - \frac{d_{OT,t_a}}{\sum_{c \mid x_{c,t_a} > 0} d_{OT,t_a}}$	$p(t_b) = 1 - \frac{d_{OT,t_b}}{\sum_t d_{OT,t}}$
ShiftIC				
Same as ShiftOT	D	$t_a \mid o_{IC,t_a} > 0$	$t_b \mid u_{IC,t_b} > 0$	$c_a \mid x_{ca,t_a} > 0$
	R	$p(t_a) = \frac{o_{IC,t_a}}{\sum_t o_{IC,t}}$	$p(t_b) = \frac{u_{IC,t_b}}{\sum_t u_{IC,t}}$	$p(c_a)$ uniform

three columns provide the elements that are successively picked and their selection conditions and probabilities $p(\cdot)$.

For instance, in the deterministic version of CategoryExchange all periods with over utilization of all resources are candidates as a first period t_a whereas in the random version a first day is selected at random proportionately to the over utilization on that day. The second element, t_b , is selected in a similar way by considering this time under utilization of all resources. The last element refers to the selection of a category. Any category with a number of planned patients on t_a greater than that of t_b is considered in the deterministic version. In the random version, a selection probability $p(c_a)$ is computed according to the category frequency where category c_a must meet the previous condition (used in the deterministic version), so rows D and R in the table are not separated by a line.

In the computation of probabilities for the two swaps and ShiftOT, we use notation $d_{r,t}$ to designate the target deviation $|q_{r,t} - A_{r,t}|$ where $q_{r,t}$ is the updated consumption of resource r on day t after performing the modification in the planning as required by the operator. It should be noted that in SwapIC, the consumption of beds in the ICU may change only l days later than t when the probabilities of occupying a bed for categories c_a and c_b start to be different. In all periods where a patient is added, the OT capacity constraint is checked beforehand.

4. Computational study

Several instances of the tactical planning problem were solved using CPLEX, R-VNS and FD-VNS. These instances and parameters

setting for CPLEX and the VNS are described in Section 4.1. Computational results are presented in Section 4.2.

4.1. Instances generation and parameters setting

We used the data of the Thorax Center available in the paper of Adan et al. [14] with the same values for all parameters ($N = 8, T = 28, \dots$) that are given in Appendix A.1. The basic situation is provided by the following initial values for the target throughput of patients in each of the 8 categories, $V^{\text{init}} = \{8, 10, 67, 13, 3, 2, 1, 7\}$ with a total of 111 patients and weights for the 4 resources $g^{\text{init}} = \{8, 10, 3, 5\}$ as defined by the stakeholders of the hospital. From this basic situation, we generated instances by varying the total number of patients V and target throughput values $\{V_c\}_{c=1..N}$ and also by considering several sets of weights $\{g_r\}_{r \in \{OT, IC, NH, MC\}}$ as detailed in Appendix A.2.

The total number of patients V was incrementally increased from 112 to a maximum of 123, which represents a realistic patient increase of 10%. For each of these 12 V -values, we performed 3 replications of uniform draws of target throughput values for each category c , V_c , between $0.95V_c^{\text{init}}$ and $1.15V_c^{\text{init}}$ with $c = 1..8$ until we got the desired total number of patients. The resulting target throughput values are displayed in Table A5. We thus obtained $1 + 3 \times 12 = 37$ problems out of which we selected 7 hard to solve ones as will be explained in Section 4.2.

For these 7 problems, we considered 4 sets of weights values to describe several assessments of resources flexibility, keeping the following order $g_{IC} > g_{OT} > g_{NH} > g_{MC}$ as required by the stakeholders. Chosen values for the weights are displayed in Table A6. A low flexibility of all resources (weight set #1) is a situation where

all weights take a large value. This is the opposite for the high flexibility case (weight set #2). Weight sets #3 and #4 combine low (resp. high) flexibility for OT and IC and high (resp. low) flexibility for MC and NH. In this way we obtained 28 additional problems, so we solved all in all $37 + 28 = 65$ instances.

Algorithms R-VNS and FD-VNS were coded in C and the experiments were executed on an Intel Xeon E5–2430 with a 2.20 gigahertz (16 Cores) CPU and 49 gigabytes RAM running the GNU/Linux Debian 7 operating system. We set a runtime limit for CPLEX to 1800 s throughout.

Parameter k_{max} of the VNS which represents the maximum number of deletions in the shaking procedure was set to $\alpha = 20\%$ of the total number of patients V . This percentage preserves a satisfactory trade-off between intensification and diversification of search as pilot runs showed that the shaking procedure led to improvements only when deletions were about 5% of V . Parameter m_{max} related to the number of applications of operators in R-VNS was set to 1000.

4.2. Results analysis

The first 37 problems were solved using CPLEX, R-VNS and FD-VNS. Each VNS was run 10 times with different random seeds (set from 1 to 10) and a time limit of 180 s per run. It thus takes $10 \times 180 = 1800$ s to get the minimum objective function value out of the 10 runs which makes this minimum truly comparable with the objective value provided by CPLEX as we set CPLEX execution time to 1800 s. Initial solutions for both VNS were generated at random using the procedure described in Section 3.1.

Results are displayed in Table 5 where the first two columns describe the instances with the total number of patients (V) and the replication of target values (T. rep) displayed in Table A5. We reported the gap to best bound provided by CPLEX at termination (Gap to BB %) which is defined as the ratio of the best integer solution found at termination and the best bound obtained by taking the minimum of the optimal objective values of all of the leaf nodes. Gap to best bound is zero for the first two problems as CPLEX found the optimal solution. The next columns report for the two versions of the VNS the best deviation (Min. dev. %) and the average deviation (Avg dev. %) of the objective value they provided compared with that of CPLEX.

Table 5 shows that R-VNS beat CPLEX for 19 problems out of 37 based on the minimum deviation, with best improvements of the objective function value above 2%. Over all cases, the VNS produced solutions that deviates only of 0.88% on average from that of CPLEX. The performance of FD-VNS was poor, with only one better solution than CPLEX and an average deviation of 7%. Simulations thus showed the superiority of the random local search included in R-VNS over the first descent strategy adopted in FD-VNS. Consequently FD-VNS was abandoned in the next experiments. Bold numbers in the table indicate the 7 selected instances for which R-VNS provided the best results. These instances are also those with high CPLEX gaps to best bound, showing the inability of CPLEX to reach high quality solutions.

To assess further the performance of R-VNS, we tested three values of the maximum number of deletions in the shaking procedure $k_{max} = \alpha V$ by setting α to 10%, 20% and 30% of the total number of patients for the 7 hard to solve problems. Again R-VNS was executed 10 times leading to a sample of 70 objective function values per α -value. Table 6 provides the minimum and average deviations relative to CPLEX.

Using a Friedman test for paired samples, we could hardly accept the null hypothesis of equality since the critical probability was 5.92%. Admittedly, from the overall average deviation and average of minimum deviations in Table 6, it can be observed that $\alpha = 0.30$ is the least favorable value to the performance of R-VNS. However a Wilcoxon test performed on the samples of objective values

Table 5

Performance of the VNS compared with CPLEX on 37 problems.

Instance	T. rep.	CPLEX		R-VNS		FD-VNS	
		1800 s		180 s 10 runs		180 s 10 runs	
		Gap to		Dev. (%)		Dev. (%)	
# pat. (V)		BB (%)		Min	Avg	Min	Avg
111	Init.	0.00	0.00	0.00	0.17	0.60	1.56
112	1	0.00	0.00	0.00	0.66	1.35	2.46
	2	0.04	0.16	0.16	1.23	2.30	4.06
	3	0.08	0.86	0.86	1.56	2.08	4.27
113	1	0.74	0.49	0.49	1.84	5.08	7.07
	2	1.01	0.10	0.10	1.06	1.31	5.35
	3	1.74	0.46	0.46	1.96	4.05	7.45
114	1	1.51	0.05	0.05	1.13	4.80	7.31
	2	3.44	0.03	0.03	1.71	5.10	7.03
	3	4.03	0.83	0.83	2.07	4.50	8.42
115	1	5.71	−0.74	0.53	2.31	5.99	
	2	8.97	−1.01	1.07	2.70	7.17	
	3	16.63	−1.25	0.52	3.60	7.64	
116	1	14.69	−2.01	−0.94	3.06	6.98	
	2	17.37	−1.82	−0.29	3.29	10.77	
	3	21.27	0.04	1.90	2.78	10.95	
117	1	7.76	0.03	1.20	4.26	8.15	
	2	14.07	0.22	2.69	4.44	13.73	
	3	14.27	−0.65	0.65	2.09	5.98	
118	1	17.40	0.36	1.51	4.35	12.49	
	2	21.48	−1.35	−0.47	0.97	7.17	
	3	22.18	−0.94	0.54	2.58	8.09	
119	1	8.97	−0.63	0.57	1.22	5.79	
	2	20.54	−1.41	−0.46	0.77	5.19	
	3	24.21	−0.64	1.31	4.27	7.59	
120	1	5.68	−0.11	1.52	3.26	9.76	
	2	6.18	0.63	2.90	8.57	13.35	
	3	17.36	−2.50	−0.40	−0.09	3.40	
121	1	6.45	−0.29	0.75	1.76	7.92	
	2	13.93	−2.22	−0.51	0.81	4.13	
	3	15.83	−2.24	−0.73	0.41	4.29	
122	1	4.12	−0.05	1.20	2.94	8.37	
	2	4.86	0.38	1.85	1.67	7.30	
	3	10.57	−1.48	−0.43	0.12	5.16	
123	1	4.15	0.35	0.68	2.81	7.21	
	2	6.61	0.21	0.92	0.83	4.04	
	3	6.89	−0.03	1.07	2.36	5.63	
Overall			−0.44	0.88	2.68	7.00	

obtained under $\alpha = 0.10$ and $\alpha = 0.20$ led to a critical probability of 94.64%. We can thus conclude that setting α to 10 or 20% has no impact on the performance of R-VNS. Thus, to a certain extent, R-VNS is pretty robust with regard to parameter α of the shaking procedure. We kept $\alpha = 0.20$ for the remaining experiments.

Next we considered 28 additional problems based on the same 7 hard to solve problems and the 4 sets of weights displayed in Table A6. Our intent was to assess the impact on the performance of R-VNS of extending the execution time beyond 180 s and starting from a good quality initial solution. Initial solutions were obtained by our random procedure (see Section 3.1) and by running CPLEX for 10 s. Again we performed 10 runs of R-VNS with different random seeds which led to 280 initial solutions whereas CPLEX produced a single initial solution for each of the 28 problems. On average, CPLEX initial objective function values were 50% lower than that of random initial solutions. For these 10 runs, the time limit of R-VNS was again set to 180 s under the 2 types of initial solutions. For each of the 28 problems, we then selected the random seed leading to the best and the worst objective function values after 180 s. For these particular seeds, we ran R-VNS for 1800 s. Table 7 displays the results. Each instance is described by a triplet (Total number of patients, Target replication, Set of weights). The second row provides CPLEX gaps to best bound. For each type of initial solution (random and CPLEX run for 10 s), we

Table 6
Performance of R-VNS with 3 different $k_{\max}$ -values on 7 hard to solve problems.

Instance # pat. V	T. rep.	CPLEX	R-VNS		180 s. 10 runs			
		1800 s. Gap to	$k_{\max} = 0.2V$ Dev. (%)		$k_{\max} = 0.1V$ Dev. (%)		$k_{\max} = 0.3V$ Dev. (%)	
		BB (%)	Min	Avg	Min	Avg	Min	Avg
115	3	16.63	−1.25	0.52	−1.00	0.20	−0.22	0.59
116	1	14.69	−2.01	−0.94	−2.10	−0.68	−1.66	−0.66
117	3	14.27	−0.65	0.65	−0.86	0.50	−1.06	0.50
118	2	21.48	−1.35	−0.47	−1.63	−0.62	−1.35	0.06
119	2	20.54	−1.41	−0.46	−2.67	−0.55	−1.76	−0.22
120	3	17.36	−2.50	−0.40	−2.47	−0.19	−1.70	−0.34
121	3	15.83	−2.24	−0.73	−2.24	−0.40	−1.97	−0.47
		Overall	−1.63	−0.26	−1.85	−0.25	−1.39	−0.08

provide the average, minimum and maximum deviations of R-VNS objective values after 180 s compared with the CPLEX solution after 1800 s. Columns labeled Best and Worst display the deviation to the CPLEX solution of the objective function value of R-VNS after 1800 s for the most and the least promising random seed value.

Random initialization: Simulation results show that R-VNS outperforms CPLEX in all cases but 3, based on minimum deviations out of 10 runs after 180 s (4th column). In more than 1/3 cases, minimum deviations are between −2% and more than −3%. Furthermore, with an overall average deviation of −0.06% R-VNS performs even better on these 28 problems than on the 37 problems. Starting from the most promising random seed (which provides the minimum deviation for each problem), prolonging

the execution time from 180 s to 1800 s led to further improvements in half cases with a maximum improvement of about −1% and a median improvement of 0.03% only. In half cases as well, the last improvement of the objective function value took place for execution times below 180 s. With the worst random seeds however, prolonging the search time always led to substantial improvements of the solutions (comparison between columns 'Max' and 'Worst'), up to −3.68% and a median value of about −2%. But with an average overall deviation of −0.49%, it seems definitely a better choice to run 10 times R-VNS with 180 s from which we select the minimum, as the average of the minimum deviations reaches a better score of −1.45% for comparable execution times. Put another way, running R-VNS for 1800 s only once with an arbitrary and unlucky random seed leads to an expected deviation of −0.49% whereas running R-VNS 10 times with 10 different seeds for 180 s, allows for an expected improvement of −1.45%.

CPLEX initialization: With a time limit of 180 s and 10 runs out of which we take the minimum deviation, R-VNS outperforms CPLEX for 22 problems, and an average of minimum deviations of −1.17%. The performance of R-VNS here is not as good as its performance under random initialization. Also, extending the execution time to 1800 s for the best seeds led to an average deviation of −1.30% which is less than the average of −1.59% we got under random initialization (comparison of columns labeled 'Worst'). The only advantage of starting from a good initial solution provided by CPLEX is to get a better overall average deviation (−0.22% < −0.06%) and a better overall deviation for the worst seeds (−0.77 < −0.49). This means that with a good starting solution, R-VNS reaches less worse but also less good solutions.

Table 7
Performance of the R-VNS compared with CPLEX on 28 additional problems.

Instance	CPLEX gap (%)	R-VNS random initial solution					R-VNS CPLEX initial solution				
		180 s 10 runs – dev. %			1800 s – dev. %		180 s 10 runs – dev. %			1800 s – dev. %	
		Avg	Min	Max	Best	Worst	Avg	Min	Max	Best	Worst
(115,3,1)	14.68	2.12	0.45	4.36	0.00	1.04	1.60	0.51	3.00	0.51	0.00
(115,3,2)	16.88	0.78	−0.81	3.42	−0.89	2.77	2.36	0.60	3.89	0.60	1.63
(115,3,3)	16.87	−0.38	−1.49	0.43	−1.55	−2.04	−0.78	−1.33	−0.22	−1.33	−2.37
(115,3,4)	15.75	1.09	−0.09	2.07	−0.32	1.62	0.77	0.36	1.46	0.18	0.36
(116,1,1)	12.78	0.65	−0.31	2.44	−0.45	0.01	0.68	−0.13	2.05	−0.13	1.45
(116,1,2)	15.16	−1.00	−2.54	0.86	−2.64	−2.35	−1.38	−2.21	−0.74	−2.64	−2.64
(116,1,3)	13.88	−0.09	−1.99	1.39	−1.99	−1.36	−1.15	−1.34	−0.87	−1.42	−1.42
(116,1,4)	14.58	−0.78	−1.55	0.52	−1.64	−1.27	−0.74	−1.33	0.74	−1.33	−1.33
(117,3,1)	12.88	0.68	0.03	1.75	0.03	0.50	1.69	0.18	3.00	0.67	0.68
(117,3,2)	15.45	−0.63	−1.63	0.53	−1.63	−1.24	−0.70	−2.17	0.35	−2.32	−0.97
(117,3,3)	13.65	0.00	−1.01	1.28	−1.09	0.75	0.53	−0.59	1.43	−0.59	1.43
(117,3,4)	14.77	0.04	−1.56	1.76	−1.56	0.12	0.15	−0.73	1.44	−1.40	−0.44
(118,2,1)	19.51	−0.59	−2.27	0.82	−2.27	−2.10	−0.83	−1.87	0.49	−1.87	−0.71
(118,2,2)	24.98	−1.50	−2.77	0.59	−2.77	−1.55	−2.01	−3.11	−0.97	−3.11	−2.77
(118,2,3)	20.64	−0.12	−1.04	0.47	−1.04	−0.89	−0.30	−1.45	1.37	−1.93	−1.04
(118,2,4)	18.65	1.21	−0.01	3.49	−0.07	0.07	1.17	0.19	2.24	−0.12	0.57
(119,2,1)	19.61	−0.57	−3.24	1.40	−3.24	−1.02	−0.36	−1.78	1.00	−3.17	−1.40
(119,2,2)	24.05	−0.36	−1.66	1.06	−1.66	−1.26	0.48	−1.02	1.69	−1.02	−0.90
(119,2,3)	22.96	−0.88	−2.21	0.72	−2.21	−2.14	−0.73	−2.18	−0.13	−2.43	−1.52
(119,2,4)	20.00	0.23	−1.82	1.82	−1.82	0.66	−0.08	−0.90	0.85	−0.90	−0.90
(120,3,1)	14.90	0.01	−0.76	1.04	−1.47	0.53	−1.01	−2.08	0.28	−2.08	−1.11
(120,3,2)	18.48	−0.13	−1.19	0.56	−1.19	−2.75	−1.25	−1.54	−0.87	−1.63	−1.01
(120,3,3)	18.78	−1.33	−2.38	−0.15	−2.38	−2.09	−1.92	−2.54	−0.56	−2.54	−1.83
(120,3,4)	15.06	−1.05	−2.63	0.41	−2.82	−0.38	−1.83	−2.35	−0.61	−2.43	−0.65
(121,3,1)	11.45	2.40	0.29	4.66	−0.21	0.98	2.51	1.91	3.24	1.91	0.60
(121,3,2)	16.68	−1.23	−3.11	0.75	−3.11	−2.12	−1.98	−2.54	−1.10	−2.54	−2.65
(121,3,3)	15.12	0.12	−1.27	1.38	−1.56	0.95	0.11	−1.13	0.71	−1.16	−0.86
(121,3,4)	14.30	−0.28	−1.98	1.87	−2.97	0.79	−1.06	−2.21	0.50	−2.21	−1.69
Overall		−0.06	−1.45	1.49	−1.59	−0.49	−0.22	−1.17	0.84	−1.30	−0.77

5. Conclusion

In this paper we have proposed a Variable Neighborhood Search algorithm for the tactical planning problem of surgeries, that belongs to the general class of master surgery scheduling problems. Results of extensive simulation experiments based on the functioning of a cardiothoracic surgery center in the Netherlands show the ability of our approach to provide high quality solutions in fast running times, especially for hard-to-solve problems for which the VNS beat CPLEX in almost all cases, with improvements of the objective function often in the range 2–3%. The very limited number of parameters to set and the algorithmic simplicity make our approach quite suitable for being implemented in a decision support system for hospitals. The approach is therefore promising to be adapted to other specialities in a hospital. In a near future, our intent is to implement a similar approach to the planning of an orthopedic center in Paris.

Appendix A. Test data

A.1. Data of the Thorax center

Based on historical data of surgeries in the Thorax Center

Table A1
Patient groups, OT hours and pre-operative days.

Patient group (c), $N = 8$	Example of procedures	OT hours (s_c)	Pre-op. day (l_c)
(1) Child, simple	Closure ventricular septal defect	4	0
(2) Child, complex	Arterial switch	8	0
(3) Adult, short OT, short IC	Coronary bypass (CABG)	4	1
(4) Adult, long OT, short IC	Mitral valve plasty	8	1
(5) Adult, short OT, middle IC	CABG with expected medium IC	4	1
(6) Adult, long OT, middle IC	Heart transplant	8	1
(7) Adult, long OT, long IC	Thoraco-abdominal aneurysm	8	1
(8) Adult, very short OT, no IC	Cervical mediastinoscopy	2	1

Table A2
Capacity and target utilization levels for the resources.

Day (t)	OT hours		IC beds		MC beds		IC nursing hours	
	Capacity ($K_{OT,t}$)	Target ($A_{OT,t}$)	Capacity ($K_{IC,t}$)	Target ($A_{IC,t}$)	Capacity ($K_{MC,t}$)	Target ($A_{MC,t}$)	Capacity ($K_{NH,t}$)	Target ($A_{NH,t}$)
Monday	36	29	10	7	36	27	133	91
Tuesday	36	29	10	7	36	27	133	91
Wednesday	36	29	10	7	36	27	133	91
Thursday	36	29	10	7	36	27	133	91
Friday	29	25	10	7	36	27	133	91
Saturday	0	0	8	2	36	27	52	26
Sunday	0	0	8	2	36	27	52	26

Table A3
Probabilities $p_{ICU,c,j}$ for all $c = 1, \dots, 8$ and for all $j = 0, \dots, L_{ICU}^{\max}$.

Patient group (c), $N = 8$	Day									
	0	1	2	3	4	5	6	7	8	9
(1) Child, simple	0.93	0.07	0.05	0.02	0	0	0	0	0	0
(2) Child, complex	1	0.09	0.02	0	0	0	0	0	0	0
(3) Adult, short OT, short IC	0.99	0.16	0.05	0.02	0.01	0.01	0.01	0	0	0
(4) Adult, long OT, short IC		0.19	0.09	0.04	0.04	0.03	0.01	0.01	0.01	0.01
(5) Adult, short OT, middle IC		0.2	0.13	0.07	0.07	0.07	0.07	0	0	0
(6) Adult, long OT, middle IC		1	0.86	0.43	0.29	0.14	0.14	0.14	0	0
(7) Adult, long OT, long IC		L	1	1	1	1	1	0	0	0
(8) Adult, very short OT, no IC	0.21	0	0	0	0	0	0	0	0	0

Rotterdam, patients were grouped in $N=8$ categories. Table A1 provides for each patient category, examples of surgical procedures performed in the center and the expected number of hours to operate on one patient in each category c (s_c). In the last column, we give the number of pre-operative days for each category (l_c).

For each resource r , there exists a maximum available capacity per day t , $K_{r,t}$ and a target utilization level $A_{r,t}$ usually set between 70% and 80% of the capacity. These values are displayed in Table A2 and apply to every week in the 4-week planning cycle ($T = 28$).

There are 4 operating theaters available 9 h a day during the week days. The Intensive Care Unit (ICU) has 10 available beds throughout the working week and 8 beds during the week end. The Medium Care Unit (MCU) has 36 beds available every day. The available ICU nursing staff is matched with the number of beds in the ICU, except on the week ends on which the staff is reduced.

The nursing hours in the ICU $\{w_{c,t}\}$ required per day for patients in categories 1–7 are estimated to 12 h throughout their stay in the ICU unless for the second day spent in the ICU for categories 5, 6 and for the second and third day for category 7 for which the needs are 24 h. Patients in category 8 require only 3 h of ICU nursing per day whatever their length of stay in the ICU. The longest stay in the ICU equals 10 days ($L_{ICU}^{\max} = 9$).

Table A4Probabilities $p_{MCU,c,j}$, for all $c = 1, \dots, 8$ and for all $j = 0, \dots, L_{MCU}^{\max}$.

Patient group (c), N = 8	Day													
	0	1	2	3	4	5	6	7	8	9	10	11	12	13
(1) Child, simple	0.05	0.26	0.26	0.26	0.26	0.21	0.14	0.07	0	0	0	0	0	0
(2) Child, complex	0	0.13	0.17	0.17	0.17	0.17	0.17	0.13	0.09	0.08	0.06	0.06	0.06	0.06
(3) Adult, short OT, short IC	0.01	0.84	0.94	0.97	0.95	0.66	0.42	0.29	0.21	0.15	0.11	0.09	0.07	0.05
(4) Adult, long OT, short IC	0	0.78	0.88	0.93	0.93	0.84	0.68	0.53	0.35	0.25	0.22	0.19	0.18	0.13
(5) Adult, short OT, middle IC	0	0.8	0.87	0.93	0.93	0.87	0.8	0.6	0.6	0.4	0.33	0.33	0.13	0.07
(6) Adult, long OT, middle IC	0	0	0.14	0.57	0.71	0.86	0.86	0.86	1	1	0.86	0.86	0.86	0.86
(7) Adult, long OT, long IC	0	0	0	0	0	0	0	1	1	1	1	1	1	1
(8) Adult, very short OT, no IC	0.64	0.54	0.41	0.26	0.13	0.1	0.08	0.08	0.03	0.03	0	0	0	0

Patient group (c), N = 8	Day													
	14	15	16	17	18	19	20	21	22	23	24	25	26	27
(1) Child, simple	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(2) Child, complex	0.06	0.04	0.02	0.02	0.02	0	0	0	0	0	0	0	0	0
(3) Adult, short OT, short IC	0.04	0.02	0.02	0.01	0.01	0.01	0.01	0	0	0	0	0	0	0
(4) Adult, long OT, short IC	0.1	0.06	0.06	0.06	0.04	0.04	0.04	0.03	0.03	0.01	0.01	0.01	0.01	0.01
(5) Adult, short OT, middle IC	0.07	0.07	0	0	0	0	0	0	0	0	0	0	0	0
(6) Adult, long OT, middle IC	0.71	0.71	0.71	0.71	0.57	0.57	0.57	0.43	0.43	0.29	0.14	0	0	0
(7) Adult, long OT, long IC	1	1	1	0	0	0	0	0	0	0	0	0	0	0
(8) Adult, very short OT, no IC	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A5Values V_c of target throughput of patients in each category c , for each total number of patients and each replication.

Instance		Category (c)							
Nb pat.	Target rep.	1	2	3	4	5	6	7	8
111	Init.	8	10	67	13	3	2	1	7
112	1	8	10	66	14	3	2	1	8
	2	8	11	67	13	3	2	1	7
	3	8	10	67	13	4	2	1	7
113	1	10	10	64	15	4	2	1	7
	2	8	12	67	13	3	2	1	7
	3	8	10	67	15	3	2	1	7
114	1	8	12	67	13	3	2	1	8
	2	8	11	69	13	3	2	1	7
	3	8	11	68	13	4	2	1	7
115	1	8	11	68	13	4	2	1	8
	2	8	10	70	14	3	2	1	7
	3	9	11	67	14	3	3	1	7
116	1	9	10	69	15	3	2	1	7
	2	8	11	70	13	3	2	2	7
	3	8	11	69	13	4	3	1	7
117	1	8	12	67	15	3	3	2	7
	2	8	10	70	13	3	3	2	8
	3	8	12	67	15	4	2	1	8
118	1	8	10	70	15	4	3	1	7
	2	9	11	68	14	4	3	1	8
	3	8	11	72	13	4	2	1	7
119	1	8	11	71	14	3	3	2	7
	2	8	10	74	14	3	2	1	7
	3	9	9	73	14	3	2	1	8
120	1	9	10	71	15	4	2	2	7
	2	10	10	74	12	3	2	2	7
	3	8	12	72	15	3	2	1	7
121	1	9	10	72	15	3	3	2	7
	2	10	11	73	13	4	2	1	7
	3	9	11	73	15	3	2	1	7
122	1	8	11	73	15	3	2	2	8
	2	8	10	76	13	4	3	1	7
	3	8	10	75	15	4	2	1	7
123	1	9	10	76	13	4	2	2	7
	2	9	11	70	15	4	3	2	9
	3	8	11	77	13	4	2	1	7

We use a stochastic length of stay in the ICU and in the MCU, based on empirical data. Table A3 provides the probability $p_{ICU,c,j}$ that a patient in category c is in the ICU j days after surgery,

$j = 0, \dots, L_{ICU}^{\max}$, with $L_{ICU}^{\max} = 9$. For instance, any patient in category 8 has 21% of chance to be in the ICU zero days after surgery, that is the day of surgery ($p_{ICU,8,0} = 0.21$). In such a case, the patient will

Table A6Different sets of values for g_r , the absolute weight for each resource r and corresponding relative values α_r .

Weight set #	Type	Absolute weight g_r				Relative weight α_r			
		OT	IC	MC	NH	OT	IC	MC	NH
0	Initial	8	10	3	5	0.1674	0.7566	0.0468	0.0291
1	Low Flex.	9	10	4	6	0.1807	0.7259	0.0599	0.0335
2	High Flex.	7	10	2	4	0.1530	0.7901	0.0326	0.0243
3	Med. Flex.	9	10	2	4	0.1884	0.7570	0.0312	0.0233
4	Med. Flex.	7	10	4	6	0.1464	0.7563	0.0624	0.0349

only stay overnight in the ICU.

Table A4 provides the same information type for the MCU. For instance, $p_{MCU,6,23} = 0.29$ means that any patient in category 6 has 29% chance to be in the MCU 23 days after surgery. We have $L_{MCU}^{\max} = 27$.

A.2. Data of instances

Values of target throughput of patients $\{V_c\}_{c=1..8}$ are displayed in Table A5 for each total number of patients (first column labeled “Nb pat.”) and for each replication (second column labeled “Target rep.”). The first row provides the basic situation in the Thorax Center.

Table A6 displays the initial set of weights values used in the basic situation as well as the 4 additional sets of values we considered for the 7 hard to solve problems selected in Section 4.2. We also provide in the table the corresponding relative weights, as computed using Eq. (2) with values of target utilization levels $\{A_{r,j}\}$ from Table A2.

References

- [1] Choi S, Wilhelm W. On capacity allocation for operating rooms. *Comput Oper Res* 2014;44:174–84.
- [2] Ma G, Demeulemeester E. A multilevel integrative approach to hospital case mix and capacity planning. *Comput Oper Res* 2013;40:2198–207.
- [3] Gupta D. Surgical suites' operations management. *Product Oper Manag* 2007;16:689–700.
- [4] Aringhieri R, Landa P, Soriano P, Tanfani E, Testi A. A two level metaheuristic for the operating room scheduling and assignment problem. *Comput Oper Res* 2015;54:21–34.
- [5] Fei H, Chu C, Meskens N, Artiba A. Solving surgical cases assignment problem by solving surgical cases assignment problem by a branch-and-price approach. *Int J Product Econ* 2008;112:96–108.
- [6] Xiang W, Yin J, Lim G. An ant colony optimization approach for solving an operating room surgery scheduling problem. *Comput Ind Eng* 2015;85:333–45.
- [7] Cardoen B, Demeulemeester E, Beliën J. Optimizing a multiple objective surgical case sequencing problem. *Int J Product Econ* 2009;119:354–66.
- [8] Cardoen B, Demeulemeester E, Beliën J. Sequencing surgical cases in a day-care environment: an exact branch-and-price approach. *Comput Oper Res* 2009;36:2660–9.
- [9] Hans E, Houdenhoven MV, Hulshof P. A framework for healthcare planning and control. In: Hall R, Hillier F, editors. *Handbook of healthcare system scheduling*. International series in operations research and management science, vol. 168. US: Springer; 2012. p. 303–20 [chapter 12].
- [10] Cardoen B, Demeulemeester E, Beliën J. Operating room planning and scheduling: a literature review. *Eur J Oper Res* 2010;201:921–32.
- [11] Guerriero F, Guido R. Operational research in the management of the operating theatre: a survey. *Health Care Manag Sci* 2011;14:89–114.
- [12] Abdelrasol Z, Harraz N, Eltawil A. Operating room scheduling problems: a survey and a proposed solution framework. *Trans Eng Technol* 2014;717–31.
- [13] Houdenhoven MV, Oostrum JV, Hans E, Wullink G, Kazemier G. Improving operating room efficiency by applying bin-packing and portfolio techniques to surgical case scheduling. *Anesth Analgesia* 2007;105:707–14.
- [14] Adan I, Bekkers J, Dellaert N, Vissers J, Yu X. Patient mix optimization and stochastic resource requirements: a case study in cardiothoracic surgery planning. *Health Care Manag Sci* 2009;12:129–41.
- [15] Hulshof J, Boucherie R, Hans E, Hurink J. Tactical resource allocation and elective patient admission planning in care processes. *Health Care Manag Sci* 2013;16:152–66.
- [16] Beliën J, Demeulemeester E. A branch-and-price approach for integrating nurse and surgery scheduling. *Eur J Oper Res* 2008;189:652–68.
- [17] Beliën J, Demeulemeester E, Cardoen B. A decision support system for cyclic master surgery scheduling with multiple objectives. *J Sched* 2009;12:147–61.
- [18] Fügener A, Hans E, Kolisch R, Kortbeek N, Vanberkel P. Master surgery scheduling with consideration of multiple downstream units. *Eur J Oper Res* 2014;239:227–36.
- [19] Adan I, Bekkers J, Dellaert N, Jeunet J, Vissers J. Improving operational effectiveness of tactical master plans for emergency and elective patients under stochastic demand and capacitated resources. *Eur J Oper Res* 2011;213:290–308.
- [20] Mladenovic N, Hansen P. Variable neighbourhood search. *Comput Oper Res* 1997;24(11):1097–100.
- [21] Hansen P, Mladenovic N. Variable neighborhood search. In: Burke E, Kendall G, editors. *Search methodologies*. US: Springer; 2005. p. 211–38 [chapter 8].
- [22] Lim G, Mobasher A, Kardar L, Cote M. Nurse scheduling. In: Hall R, editor. *Handbook of healthcare system scheduling*. International series in operations research and management science, vol. 168. US: Springer; 2012. p. 31–64 [chapter 3].
- [23] Burke E, Li J, Qu R. A hybrid model of integer programming and variable neighbourhood search for highly-constrained nurse rostering problems. *Eur J Oper Res* 2010;203:484–93.
- [24] Dellaert N, Cayiroglu E, Jeunet J. Assessing and controlling the impact of hospital capacity planning on the waiting time. *Int J Product Res* 2016. <http://dx.doi.org/10.1080/00207543.2015.1051668> (Forthcoming).
- [25] Creemers S, Lambrecht M. Modeling a hospital queueing network. In: Boucherie R, van Dijk N, editors. *Queueing networks—a fundamental approach*. International series in operations research and management science, vol. 154. US: Springer; 2011. p. 767–98 [chapter 18].