J Comb Optim (2015) 30:982–995
DOI 10.1007/s10878-015-9860-3

Resource-constrained machine scheduling with machine eligibility restriction and its applications to surgical operations scheduling

Shan Wang[1] · **Huiqiao Su**[1] · **Guohua Wan**[1]

Published online: 3 April 2015

Abstract We study a problem arising from surgical operations scheduling and model it as a resource-constrained machine scheduling problem with machine eligibility restriction to minimize the makespan. By decomposing the problem into two sub-problems, we develop effective heuristic algorithms to solve the problem. We test the proposed algorithms on randomly generated instances as well as real data set from a large hospital. The numerical results show the effectiveness and potential practical value of the models and the algorithms.

Keywords Healthcare · Surgical operations scheduling · Machine eligibility · Resource constraint · Heuristic algorithm

1 Introduction

Many different kinds of resources are needed to treat patients in hospitals. In particular, operating rooms (ORs), surgeons, anesthesiologists, nurses and expensive surgical equipment (ESE) are among those expensive and bottleneck resources in hospitals (Smith-Daniels et al. 1988). Improving the utilization of these resources is always a critical issue in hospital resource management, and better management of these

✉ Guohua Wan
ghwan@sjtu.edu.cn

Shan Wang
wangshan_731@sjtu.edu.cn

Huiqiao Su
suhuiqiao@sjtu.edu.cn

[1] Antai College of Economics and Management, Shanghai Jiao Tong University, Shanghai 200052, China

resources may increase the benefits (both tangible and intangible) for hospitals as well as patients significantly (see, e.g., Stahl et al. 2006; Magerlein and Martin 1978; Guerriero and Guido 2011; May et al. 2011).

As a kind of bottleneck resources, ORs are very expensive and their number is limited thus improving their utilization is critical to performance improvement in hospitals. Previous literature has tackled the OR scheduling problem under different settings and using different models and solution methods. For instance, Blake and Carter (2002) study the long-term or mid-term planning problem for ORs, and Beliën and Demeulemeester (2008) focus on short-term tactical scheduling of ORs. In terms of solution methods, researchers (e.g., Jebali et al. 2006; Cardoen et al. 2009) normally build mixed integer linear programming (MILP) models and then propose various solution methods based on mathematical programming theory (e.g., Lagrange relaxation, branch-and-price) or local search (e.g., genetic algorithm or tabu search).

In practice, there are two types of surgical operations, namely elective surgical operations and emergency surgical operations. For elective surgical operations, we may schedule them well in advance, while for emergency surgical operations, we have to schedule them immediately when patients arrive at the hospitals. Generally speaking, management either reserve some ORs or some time slots for emergency surgical operations (van der Lans et al. 2006; Gerchak et al. 1996). For elective surgical operations, since most information (possibly except the processing times) is known thus a schedule can be constructed before performing the operations. For processing times, some researchers assume them as deterministic variables (e.g., Pham and Klinkert 2008), while some other researchers model them as stochastic variables (e.g., Strum et al. 1999).

When surgical operations are preformed, ORs are critical resources. However, many other resources such as surgeons and equipment are needed in the process thus resource constraints are widely taken into consideration (e.g., Fei et al. 2010).

In reality, different surgical operations should be performed in different kinds of ORs thus introducing eligibility restrictions of ORs in scheduling problems. However, only a few studies takes into consideration of the processing eligibility restrictions (e.g., Zhao and Li 2014).

In this paper, we study a scheduling problem arising in Shanghai General Hospital, a large comprehensive hospital in Shanghai, China, which has 22 ORs for in-patients, around 150 surgeons, and almost 500 types of surgical operations in one of its two branches. In total, around 20,000 elective operations are performed annually. To increase the utilization of the ORs (and many other resources), it is necessary to make an optimal schedule of the surgical operations, e.g., minimizing the makespan of the schedule. Due to many different constraints and restrictions, the problem becomes rather difficult. For instance, ORs are equipped differently, causing that one type of surgical operations may only be performed in a set of very specific ORs. Also, various resources such as surgeons, nurses, anesthesiologists and special equipment, are needed to perform a surgical operation, making the problem even more difficult.

In this study, we focus on elective surgical operations scheduling, and employ machine scheduling theory to solve the problem. We assume deterministic processing times based on estimations of processing times using statistical methods (e.g., Wright et al. 1996; Strum et al. 2000) and there are good approaches to make appointments for patients under random processing times (e.g., Ge et al. 2014). Machine eligibility and resource constraints are taken into consideration in this study as well. Although the integrated model is a good approximation of real surgical OR scheduling, it is very complicated to solve it via fast exact algorithm. We thus decompose the problem into two sub-problems and develop effective and efficient algorithms to solve it.

The rest of this paper is organized as follows. In Sect. 2, we describe the problem and how it can be decomposed into two sub-problems. In Sects. 3 and 4, we analyze the computational complexity of the problem and develop heuristic algorithms to solve it. We test performance of the proposed algorithms on both randomly generated data and real data set in Sect. 5. We conclude with discussion on the future research in Sect. 6.

2 Problem description and the model

As in the literature, we regard ORs as machines and surgical operations as jobs in a machine scheduling problem. The objective is to minimize the makespan in order to increase the utilization of the ORs. Using machine scheduling theory to solve the surgical operations scheduling problem, Pham and Klinkert (2008) model the problem as a multi-mode blocking job shop (MMBJS) while Zhong et al. (2014) model the problem as a multi-machine scheduling problem, regarding ORs, surgeons, nurses, anesthesiologists and equipment as multiple machines for surgical operations. In Wang et al. (2014), they model the daily surgical operations scheduling problem as a two-stage no-wait hybrid flow-shop problem and propose a search method based on particle swarm to solve the problem.

As discussed above, there exist various constraints and restrictions on both ORs and resources for surgical operations. To well model the problem, we make the following two assumptions.

Assumption 1 (*machine eligibility*) A surgical operation (a job) j can only be performed (processed) in a specific subset $\mathbb{M}_j \subseteq \mathbb{M}$, where $\mathbb{M}$ denotes the set of all the ORs (machines).

Assumption 2 (*unit resource requirement*) There are λ types of different resources, and there is only one unit in each type of resources. Each job needs only one unit of resources for its processing.

Assumption 1 is used to model the eligibility of ORs (machine eligibility) and Assumption 2 is used to model the situation where only the most critical resource matters in the surgical operations. Notice that in a hospital, usually surgeons, nurses and anesthesiologists work as surgical teams, thus it can be regarded as unit resources.

Using three-field notation in scheduling theory (Pinedo 2012), the problem can be denoted as $P|\mathbb{M}_j, Res\,\lambda\,11|C_{max}$, and the problem can be stated as follows.

There are n jobs (surgical operations) $\{J_1, J_2, \ldots, J_n\}$ to be processed on m parallel machines (ORs) $\{M_1, M_2, \ldots, M_m\}$ and job J_j can only be processed in a subset of the machines $\mathbb{M}_j$ (machine eligibility restrictions). We use an eligibility matrix

$A_{n \times m}$ to represent these restrictions: if job J_j can be processed on M_i then $A(j, i) = 1$; otherwise $A(j, i) = 0$. The symbol $Res\,\lambda\,11$ denotes that there are λ types of resources with each type having only one unit, and it needs only one unit of resources for processing a job. The objective is to minimize C_{max}, the time when all jobs are completed.

The problem $P|\mathbb{M}_j, Res\,\lambda\,11|C_{max}$ is obviously NP-hard since many problems such as $P||C_{max}$ are its special cases, and with resource constraints and machine eligibility restrictions, it is not expected to find a fast algorithm to solve it to optimality. Hence, we decompose it into two sub-problems, namely, job assignment (to the parallel machines) and job sequencing. Decomposition is a strategy widely adopted in surgical operations scheduling (see, for instance, Guinet and Chaabane 2003; Jebali et al. 2006; Testi et al. 2007; Fei et al. 2010; Zhao and Li 2014; Zhong et al. 2014). The method is also consistent with the practices in many hospitals although an integrated solution approach is more desirable. In this study, we first determine the assignment of the jobs to eligible machines, and then sequence the jobs (which have been assigned to specific machines) taking into consideration of the resource constraints.

2.1 Assignment

In the assignment problem, we decide which jobs should be processed on which machine. The sequences of jobs are not the concern here but the machine eligibility should be dealt with. Machine eligibility restriction is well studied in the machine scheduling literature but little literature on surgical operations scheduling is concerned with it. In machine scheduling literature, Glass et al. (2007) and Ou et al. (2008) study the worst-case performance of strongly polynomial-time approximation algorithms for nested machine eligibility and inclusive machine eligibility, respectively. For general machine eligibility, Shchepin and Vakhania (2005) propose a 2-approximation polynomial-time algorithm, and Vairaktarakis and Cai (2003) examine several heuristic rules. Notice that few studies consider resource constraints simultaneously, which are important aspects of the current problem. There are several important features in this study:

– Although the resources can be shared among different machines (ORs), it is costly to move the resources from one machine to another machine.
– Centralizing the use of resources can lower the difficulty of sequencing jobs in the second stage.

Hence, it is a good strategy to assign jobs requiring the same resources on the same machine. This means each machine should use as few types of resources as possible. To clearly describe the problem, we introduce the notation $\mathbb{R}_i$, the set of resources that machine M_i uses. Then, the assignment problem becomes a bicriteria scheduling problem with machine eligibility to minimize both the makespan and the maximum resource usage by one machine, denoted as $P|\mathbb{M}_j|(C_{max}, R_{max})$, where $\mathbb{R}_i$ is the set of used resources by machine M_i, and $R_{max} = \max_i |\mathbb{R}_i|$.

2.2 Sequencing

Before sequencing the jobs, it is necessary to resolve the conflicts of resources requirements. In hospitals, there are many good practices when scheduling the surgical operations as can be summarized as follows.

- The aseptic surgical operations need to be scheduled before the non-aseptic surgical operations;
- High-priority patients need to be scheduled before normal patients;
- Surgical operations with general anesthesia need to be scheduled before surgical operations with local anesthesia;
- Surgical operations with easily estimated durations need to be scheduled before surgical operations whose durations are hard to estimate;
- Surgeons' preferences are important.

The above practices are discussed in Zhang et al. (2014). Also, Min and Yih (2010) consider the patient priority, and preference of surgeons are considered in Meskens et al. 2013 and Cardoen et al. 2009. In the current problem, we regard all of these practices as constraints. For simplicity, we make the following assumption.

Assumption 3 (*partial pre-determined sequence*) A partial sequence of jobs using the same type of resources can be pre-determined and remains unchanged in the later stage.

Although some pre-determined sequences can be changed in reality, the idea of introducing these practices in the problem is important for solving the problem. In fact, by this assumption, the resource constraint $Res\,\lambda\,11$ can be replaced by pre-determined partial sequences which formulate λ chains. Hence the sequencing problem can be transformed into a dedicated machine scheduling with chain constraints, i.e., $PD|Chains|C_{max}$, where PD represents dedicated parallel machines since the assignment has been decided already, and $Chains$ represent the partial sequences of jobs using the same resources, and each resource decides a chain of jobs using it.

2.3 MILP model

Before developing solution approach to the scheduling problem, we now describe an MILP model for the problem to help understand the complexity of the problem, and for potential base of search procedure. The notations are listed in Table 1.

Then the MILP model for $P|\mathbb{M}_j, Chain|C_{max}$ can be formulated as follows.

$$min\ \ C_{max} \tag{1a}$$

$$s.t.\ \sum_{i=1}^{m} y_{j,i} = 1 for \qquad j = 1, \ldots, n \tag{1b}$$

$$y_{j,i} \leq A_{j,i} \qquad for\ \ j = 1, \ldots, n \quad i = 1, \ldots, m \tag{1c}$$

Table 1 Notation

Notation	Explanation
Parameters	
m	Number of machines (ORs)
n	Number of jobs (surgical operations)
λ	Number of chains (resources)
M_i	Machine i
J_j	Job j
$\mathbb{M}_j$	Set of machines which can process job j
$A_{j,i}$	If job j can be processed in machine i, then it equals to 1; otherwise it equals to 0
p_j	Processing time of job j
$c(j)$	The chain where job j is
$seq_{j,k}$	If $c(j) = c(k)$ and job j is pre-determined before k, then it equals to 1; otherwise it equals to 0
M	A big number M
Decision variables	
$x_{j,k}$	if job j is sequenced before k, then it equals to 1; otherwise it equals to 0
$y_{j,i}$	If job j is assigned to machine i, then it equals to 1; otherwise it equals to 0
s_j	Starting time of job j
C_{max}	Makespan

$$x_{j,k} + x_{k,j} = 1 \qquad for \quad j = 1, \ldots, n; \; k = 1, \ldots, n; \; j \neq k \tag{1d}$$

$$x_{j,k} = seq_{j,k} \qquad for \quad j = 1, \ldots, n; \; k = 1, \ldots, n; \; c(j) = c(k) \tag{1e}$$

$$s_j \geq seq_{k,j}(s_k + p_k) \qquad for \quad k = 1, \ldots, n; \; j = 1, \ldots, n; \; c(j) = c(k) \tag{1f}$$

$$\mathrm{M}(2 + x_{k,j} - y_{k,i} - y_{j,i}) + s_k \geq p_j + s_j \qquad for \quad j = 1, \ldots, n; \; k = 1, \ldots, n; \; i = 1, \ldots, m; \; k \neq j \tag{1g}$$

$$C_{max} \geq s_j + p_j \qquad for \quad j = 1, \ldots, n \tag{1h}$$

$$x_{j,k}, \quad y_{j,i} \in \{0, 1\} \qquad for \quad j = 1, \ldots, n; \; k = 1, \ldots, n; \; i = 1, \ldots, m \tag{1i}$$

$$s_j \geq 0 \qquad\qquad for \quad j = 1, \ldots, n \tag{1j}$$

Objective (1a) is the objective function of minimizing the makespan, which is larger than or equal to the completion time of all jobs in constraint (1h). Constraint (1b) makes sure that every job is assigned to one machine. Machine eligibility constraint is embodied in constraint (1c). Constraint (1d) guarantees either J_j is before J_k or J_k is before J_j. Constraint (1e) ensures partial predetermined sequences should not be changed. For two jobs in same chain or same machine, the starting time of the later one will not be earlier than the finishing time of the former one, which are guaranteed in constraint (1f) and (1g), respectively. Constraint (1i) and (1j) define the allowed values for decision variables.

Obviously the model does not allow for an efficient algorithm to solve it to optimality, thus we turn to the decomposition strategy for solving the problem.

3 The problem $P|\mathbb{M}_j|(C_{max}, R_{max})$

It is well known that $P||C_{max}$ is NP-hard (see, e.g., Pinedo 2012), and the current problem is much more difficult than it because of the eligibility constraints and resource usage objective.

Vairaktarakis and Cai (2003) study several heuristic rules for $P|\mathbb{M}_j|C_{max}$, and they claim lowest average workload (LAW) has the best performance. The basic idea for LAW is to choose a machine with the lowest average workload first and then choose a job with the largest processing time to process on the chosen machine.

As an extension, we propose a modified LAW (MLAW) by taking resource usage into consideration. A machine is still chosen by the lowest average workload, while a job with the largest modified processing time is chosen. The modified processing time becomes much smaller if the job needs a resource which has not been used before by the chosen machine. This modification makes the resource usage by each machine smaller while still keeps the makespan. To describe the algorithms, we supplement new notations in Table 2.

We define the average workload of M_i as the summation of the assigned workload and the expected workload as follows:

Table 2 New notations for assignment

Notation	Explanation
Parameters	
$\mathbb{B}_i$	Set of available unassigned jobs for machine i
$\mathbb{R}_i$	Set of used resources by machine i
$\mathbb{UR}$	Set of used resources by all machines
mp_j	Modified processing time of job j
W_i	Expected workload of machine i
t_i	Available time (assigned workload) of machine i
Decision variable	
$m(j)$	Index of machine that has been assigned to job j

$$workload_i := t_i + \sum_{J_j \in \mathbb{B}_i} \frac{p_j}{|\mathbb{M}_j|}$$

and the modified processing time of J_j as follows:

$$mp_j := \begin{cases} p_j & c(j) \in \mathbb{R}_i \\ \dfrac{p_j}{\left(1 + \dfrac{|\mathbb{B}_i| * |\mathbb{R}_i|}{|\mathbb{UR}|}\right)} & c(j) \notin \mathbb{R}_i \end{cases}$$

When a job J_j requires a resource which has not been used by the chosen machine before, its modified processing time decreases with the number of used resources by the chosen machine and the available unassigned jobs of the chosen machine, but increases with the number of total used resources by all machines. It is easy to know that the more used resources of the chosen machine are, the worse the performance is. As to the other two aspects, the more unassigned jobs are, the more expected usage of resources by the chosen machine is; and the less total used resources is, the more unused resources are. These three reasons make job J_j less likely to be chosen. Based on these intuitions, we propose the following algorithm.

Algorithm MLAW

- Step 0: Set: $\mathbb{B}_i := \{J_j : A_{j,i} = 1\}$, $W_i := \sum_{J_j \in \mathbb{B}_i} \dfrac{p_j}{|\mathbb{M}_j|}$, $t_i := 0$;
- Step 1: Find M_{i^0} such that $\mathbb{B}_{i^0} \neq \Phi$ and $(t_{i^0} + W_{i^0}) = \min_{i=1}^{m}(t_i + W_i)$;
- Step 2: Calculate mp_j, and find J_{j^0} such that $mp_{j^0} = \max_{j=1}^{n}\{mp_j\}$;
- Step 3: Let $m(j^0) := i^0$, $t_{i^0} := t_{i^0} + p_{j^0}$; $\mathbb{R}_{i^0} := \mathbb{R}_{i^0} + \{c(j^0)\}$, $\mathbb{UR} := \mathbb{UR} + \{c(j^0)\}$;
- Step 4: For $M_i \in \mathbb{M}_{j^0}, set : W_i := W_i - \dfrac{p_{j^0}}{|\mathbb{M}_{j^0}|}$, $\mathbb{B}_i := \mathbb{B}_i - \{J_{j^0}\}$; Go to Step 1 till $\bigcup_{i=1}^{m} \mathbb{B}_i = \Phi$.

In the algorithm, Step 0 initializes the unassigned job set and expected workload of each machine. Step 1 chooses a machine with smallest total average workload; Step 2 chooses a job with largest modified processing time; Step 3 completes the assignment and changes the set of used resources by the chosen machine and the set of total used resources by all machines; Step 4 changes sets of unassigned jobs and expected workloads of machines which can process the chosen job. Repeat Steps 1–4 until that all jobs are assigned. It is easy to see that MLAW algorithm takes $O(n^2 \log n)$ time to complete the job assignment.

4 The problem $PD|Chains|C_{max}$

4.1 Computational complexity

We now face a scheduling problem with dedicated parallel machines since the assignment has been decided on the job assignment. Furthermore, partial sequences have

Table 3 New notations for sequencing

Notation	Explanation
$pre(j)$	Direct predecessor of job J_j (is *null* if job J_j is the first job in a chain)
$suc(j)$	Direct successor of job J_j (is *null* if job J_j is the last job in a chain)
$\mathbb{C}_l$	Set of jobs in chain l
r_j	Earliest starting time of job J_j, it is determined by followed jobs by J_j
q_j	Shortest remaining time of job J_j, it is determined by following jobs of J_j
$\mathbb{A}$	Set of current available jobs

been determined, which form job chains. Hence, the left task is to decide the sequences of unsequenced jobs and the problem becomes $PD|Chains|C_{max}$. Its computational complexity is unknown but in fact, $PD|Chains|C_{max}$ can be transformed to a classic job shop problem $J||C_{max}$ if we regard an entire chain as a job and jobs in a chain as tasks. $J||C_{max}$ is a strongly NP-hard problem (Pinedo 2012), thus we know that $PD|Chains|C_{max}$ is also NP-hard in strong sense.

4.2 Heuristic algorithm

The algorithms for $J||C_{max}$ can be used to solve the problem $PD|Chains|C_{max}$. However, if Assumption 3 does not hold, the sequencing problem is not a job shop problem anymore and these algorithms are of little value for solving the related problem.

To design an algorithm which can be extended to deal with other related problems, we reduce the problem $1|r_j|L_{max}$ to our problem. In fact, Jackson's rule works well as a heuristic algorithm for $1|r_j|L_{max}$ (Pinedo 2012). Following this idea, we propose longest remaining chain (LRC) rule for solving the problem $PD|Chains|C_{max}$. The idea is, each time, from all jobs which are available for processing, to choose a job whose following jobs have the largest total processing time. New notations for LRC are supplemented in Table 3.

Algorithm LRC

- Step 0: Set: $t := t_i := 0$, $r_j := r_{pre(j)} + p_{pre(j)}$, $q_j := q_{suc(j)} + p_{suc(j)}$, $\mathbb{B}_i := \{J_j : m(j) = i\}$, and $\mathbb{C}_l := \{J_j : c(j) = l\}$;
- Step 1: $\mathbb{A} := \{J_j : r_j \leq t and J_j \in \bigcup_{i=1}^m \mathbb{B}_i\}$; Find J_{j^0} such that $p_{j^0} = \max_{J_j \in \mathbb{A}}\{p_j : q_j = \max_{J_k \in \mathbb{A}}\{q_k\}\}$;
- Step 2: $s_{j^0} := \max\{t_{m(j^0)}, r_{j^0}\}$, $t_{m(j^0)} := s_{j^0} + p_{j^0}\mathbb{B}_{m(j^0)} := \mathbb{B}_{m(j^0)} - \{J_{j^0}\}$, $\mathbb{C}_{c(j^0)} := \mathbb{C}_{c(j^0)} - \{J_{j^0}\}$;
- Step 3: If $s_{j^0} > r_{j^0}$, then for $J_j \in \mathbb{C}_{c(j^0)} : r_j := r_j + s_{j^0} - r_{j^0}$;
- Step 4: Find $t := \min_{i=1}^m\{\max\{t_i,\ \min_{j\in\mathbb{B}_i}\{r_j\}\}\}$; Go to Step 1 till $\bigcup_{i=1}^m \mathbb{B}_i = \Phi$.

In the algorithm, Step 0 initializes the current time, machines' available times, jobs' earliest starting times and shortest remaining times, and set of undone jobs for each machine and each resource. Step 1 finds a job from current available jobs with largest shortest remaining time (tie broken by picking the one with largest processing time). Step 2 sets the starting time for the selected job, changes the available time for the corresponding machine, and deletes the selected job from the undone job sets. Steps 3 and 4 modifies the earliest starting times for undone jobs and current time. Repeat Steps 1–4 till all jobs are processed. It is easy to see that Algorithm LRC takes $O(n^2)$ time to complete the sequencing process.

Notice that the basic idea of Algorithm LRC is to pick a job with largest shortest remaining time from jobs which can be started now, thus it can also be used for the case where Assumption 3 does not hold.

5 Performance evaluation

5.1 Randomly generated data

To evaluate the performance of the proposed algorithms for the whole problem, we generate 49 data groups with size 10. These groups of data vary in machine number, job number, resource number and processing times. Processing times, eligibility matrix and partial sequences are generated randomly. In Table 4, we show the average performance gap (APG) and worst performance gap (WPG) between our solutions and the optimal solutions for the 49 groups of data. The APGs of all the groups are good with their values being mostly between 0.1 and 0.2, although the WPGs may be up to more than 0.5. The total average performance gap for all data is 0.1246, a very impressive result[1], where: m is the number of machines, n is the number of jobs, λ is number of resources, and p is the average processing time.

An interesting observation is that, with fixed job number and average processing time, the decomposition algorithms have better performance for the case with a few machines and a few resources. We define the tightness for the two things, the less machines and resources, the higher tightness. Figure 1a shows the impact of tightness on performance of our decomposition algorithms (tightness of group with $m = 4$ and $\lambda = 8$ is normalized to be 1). From Fig. 1b, we cannot find out how problem scale effects performance of our algorithms, but longer average processing time makes performance more stable. The problem scale is defined in such a way that, for fixed tightness, the more jobs the larger scale (problem scale of group with $n = 20$, $m = 5$ and $\lambda = 10$ is normalized to be 1). Figure 1c–f show the different performance with different the number of machines, the number of jobs, the number of resources and the average processing time, respectively. It is not obvious how these parameters effect the performance

[1] The performance measure is calculated by $\left(\frac{(Value\ of\ the\ Algorithm)}{(Optimal\ Value)} - 1\right)$. However, getting optimal values for some instances are too time-consuming, hence for such data groups we use near optimal values with 8 % relative gap. We then use $\left(\frac{(Value\ of\ the\ Algorithm) * 1.08}{(Near\ Optimal\ Value)} - 1\right)$ to calculate the APG/WPG. The APG/WPG of the data groups with mark * are calculated in this way in Table 4.

Table 4 Gap summary

m	n	λ	p	APG	WPG	m	n	λ	p	APG	WPG
16	4	8	5	0.1070	0.3182	16	4	8	10	0.0507	0.2037
16	4	8	25	0.0726	0.2045	16	4	8	50	0.1491	0.2490
16	4	8	100	0.1557	0.4202	18	5	12	5	0.0936	0.2667
18	5	12	10	0.1227	0.2857	18	5	12	25	0.1530	0.3689
18	5	12	50	0.1100	0.3405	18	5	12	100	0.0798	0.2260
20	3	6	10	0.1009	0.2727	20	3	6	100	0.1062	0.3017
20	4	8	10	0.1353	0.2813	20	4	8	100	0.1651	0.3314
20	5	10	10	0.1566	0.3276	20	5	10	100	0.1726	0.2988
20	6	12	10	0.1234	0.1887	20	6	12	100	0.1245	0.5066
24	6	12	10	0.1809	0.2909	24	6	12	100	0.1441	0.3718
26	4	8	10	0.1610	0.3411*	26	4	8	100	0.1442	0.2923*
30	3	6	10	0.1481	0.1904*	30	3	6	100	0.1715	0.2246*
30	4	8	10	0.1439	0.3114*	30	4	8	100	0.1678	0.5361*
30	5	10	10	0.1596	0.3957*	30	5	10	100	0.1464	0.2921*
30	6	12	10	0.1086	0.2697*	30	6	12	100	0.1104	0.3438*
40	10	20	10	0.0870	0.3333	40	10	20	100	0.1594	0.3824
80	20	40	10	0.1312	0.3688*	80	20	40	100	0.1418	0.2913*
20	2	8	10	0.08	0.08*	20	3	8	10	0.1479	0.2825*
20	5	8	10	0.1449	0.2778	20	6	8	10	0.1689	0.3488
20	2	4	10	0.0410	0.0745	20	3	4	10	0.1402	0.2195
20	4	4	10	0.1296	0.4127	20	5	4	10	0.0872	0.1276
20	6	4	10	0.0800	0.1728	20	2	6	10	0.0297	0.1631
20	2	10	10	0.08	0.08*	20	2	12	10	0.08	0.08*
20	4	6	10	0.1998	0.3088	20	4	10	10	0.1176	0.3333
20	4	12	10	0.0906	0.1323*						

based on only these results thus it needs more computational experiments in the future.

5.2 Real data set

We now use real data set to test our decomposition algorithms. In north branch of Shanghai General Hospital, there are 22 operating rooms and 154 surgeons in in-patient department. Surgeons in this comprehensive hospital can perform 497 different kinds of surgical operations. We collect the historical data from January 1, 2014 to July 1, 2014. During the 6 months, this branch performed a total of 9699 in-patient elective surgical operations. After data cleaning, there are 9148 available data. In working days, around 80 sets of operations were performed per day; in weekends or holidays, there were at most 20 operations being performed per day. After adding the eligibility constraint and using the realized partial sequences for the same surgeon, we implement

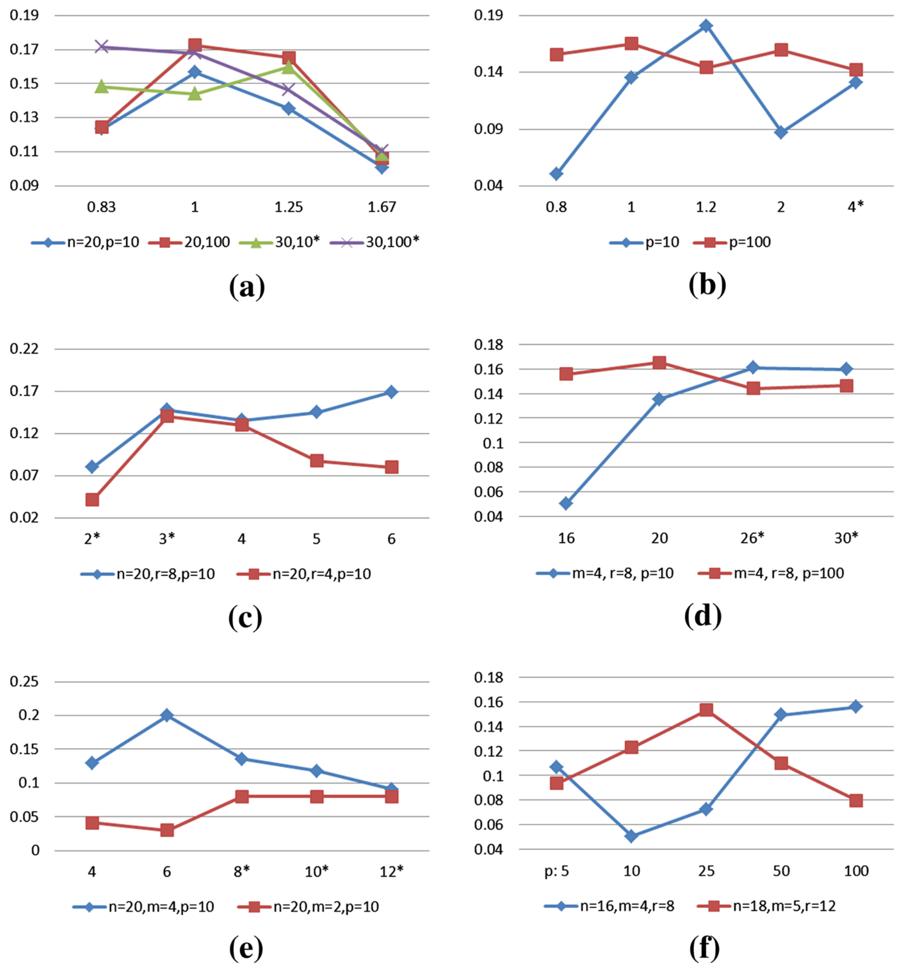

Fig. 1 **a** Effect of tightness. **b** Effect of scale. **c** Effect of machine number. **d** Effect of job number. **e** Effect of resource number. **f** Effect of processing time.

our decomposition algorithms with the historical data. The result shows an average improvement of 21.38 % for working days, and average improvement of 61.94 % for holidays, compared with the real makespan.

6 Conclusion

We study a surgical operations scheduling problem arising in hospitals. Using machine scheduling theory, we model it as a resource-constrained scheduling with machine eligibility restriction to minimize the makespan, i.e., $P|\mathbb{M}_j, Res\,\lambda\,11|C_{max}$. Due to the computational complexity of the problem, we decompose it into two sub-problems. The first sub-problem is the assignment problem, modeled as a bicriteria scheduling with machine eligibility to minimize both makespan and maximum resource usage by one machine $P|\mathbb{M}_j|(C_{max}, R_{max})$. We propose a modify LAW rule to handle the second objective R_{max}. The second sub-problem is a sequencing problem, modeled as a dedicated machine scheduling problem with chain constraint $PD|Chains|C_{max}$. We show that this problem is NP-hard in strong sense, and develop LRC algorithm to solve it. We demonstrate the effectiveness of the proposed algorithms by computational experiments. The results show that the average performance gap is 0.1246, an impressive result. By testing with real data set in Shanghai General Hospital, our algorithms achieve 21.38 % improvement for working days, and 61.94 % improvement for holidays.

For future research along this vein, we may conduct more thorough experiments to test the algorithms and implement the algorithms in decision support systems, or consider more practical constraints for the surgical operations scheduling problem.

Acknowledgments The research is supported in part by National Science Foundation of China (Grants 71125003 and 71421002).

References

Beliën J, Demeulemeester E (2008) A branch-and-price approach for integrating nurse and surgery scheduling. Eur J Oper Res 189:652–668

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

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

Fei H, Meskens N, Chu C (2010) A planning and scheduling problem for an operating theatre using an open scheduling strategy. Comput Ind Eng 58:221–230

Ge D, Wan G, Wang Z, Zhang J (2014) A note on appointment scheduling with piecewise linear cost functions. Math Oper Res 39:1244–1251

Gerchak Y, Gupta D, Henig M (1996) Reservation planning for elective surgery under uncertain demand for emergency surgery. Manage Sci 42:321–334

Glass CA, Kellerer H (2007) Parallel machine scheduling with job assignment restrictions. Naval Res Logist 54:250–257

Guerriero F, Guido R (2011) Operational research in the management of the operating theatre: a survey. Health Care Manag Sci 14:89–114

Guinet A, Chaabane S (2003) Operating theatre planning. Int J Prod Econ 85:69–81

Jebali A, Hadj Alouane AB, Ladet P (2006) Operating rooms scheduling. Int J Prod Econ 99:52–62

van der Lans M, Hans EW, Hurink JL, Wullink G, van Houdenhoven M, Kazemier G (2006) Anticipating urgent surgery in operating room departments. Tech Rep., BETA working paper WP-158

Magerlein JM, Martin JB (1978) Surgical demand scheduling: a review. Health Serv Res 13:418

May JH, Spangler WE, Strum DP, Vargas LG (2011) The surgical scheduling problem: current research and future opportunities. Prod Oper Manag 20:392–405

Meskens N, Duvivier D, Hanset A (2013) Multi-objective operating room scheduling considering desiderata of the surgical team. Decis Support Syst 55:650–659

Min D, Yih Y (2010) An elective surgery scheduling problem considering patient priority. Comput Oper Res 37:1091–1099

Ou J, Leung JYT, Li CL (2008) Scheduling parallel machines with inclusive processing set restrictions. Naval Res Logist 55:328–338

Pham DN, Klinkert A (2008) Surgical case scheduling as a generalized job shop scheduling problem. Eur J Oper Res 185:1011–1025

Pinedo ML (2012) Scheduling: theory, algorithms, and systems. Springer, New York

Shchepin EV, Vakhania N (2005) An optimal rounding gives a better approximation for scheduling unrelated machines. Oper Res Lett 33:127–133

Smith-Daniels VL, Schweikhart SB, Smith-Daniels DE (1988) Capacity management in health care services: Review and future research directions. Decis Sci 19:889–919

Stahl JE, Sandberg WS, Daily B, Wiklund R, Egan MT, Goldman JM, Isaacson KB, Gazelle S, Rattner DW (2006) Reorganizing patient care and workflow in the operating room: a cost-effectiveness study. Surgery 139:717–728

Strum DP, May JH, Vargas LG (2000) Modeling the uncertainty of surgical procedure times: comparison of log-normal and normal models. Anesthesiology 92:1160–1167

Strum DP, Vargas LG, May JH (1999) Surgical subspecialty block utilization and capacity planning: a minimal cost analysis model. Anesthesiology 90:1176–1185

Testi A, Tanfani E, Torre G (2007) A three-phase approach for operating theatre schedules. Health Care Manag Sci 10:163–172

Vairaktarakis GL, Cai X (2003) The value of processing flexibility in multipurpose machines. IIE trans 35:763–774

Wang Y, Tang J, Pan Z, Yan C (2014) Particle swarm optimization-based planning and scheduling for a laminar-flow operating room with downstream resources. Soft Comput. doi:10.1007/s00500-014-1453-z

Wright IH, Kooperberg C, Bonar BA, Bashein G (1996) Statistical modeling to predict elective surgery time: comparison with a computer scheduling system and surgeon-provided estimates. Anesthesiology 85:1235–1245

Zhao Z, Li X (2014) Scheduling elective surgeries with sequence-dependent setup times to multiple operating rooms using constraint programming. Oper Res Health Care 3:160–167

Zhong L, Luo S, Wu L, Xu L, Yang J, Tang G (2014) A two-stage approach for surgery scheduling. J Comb Optim 27:545–556