The Bi-Objective Master Physician Scheduling Problem

Aldy Gunawan • Hoong Chuin Lau

Abstract Physician scheduling is the assignment of physicians to perform different duties in the hospital timetable. In this paper, the goals are to satisfy as many physicians' preferences and duty requirements as possible while ensuring optimum usage of available resources. We present a mathematical programming model to represent the problem as a bi-objective optimization problem. Three different methods based on ε–Constraint Method, Weighted-Sum Method and Hill-Climbing algorithm are proposed. These methods were tested on a real case from the Surgery Department of a large local government hospital, as well as on randomly generated problem instances. The strengths and weaknesses of the proposed methods are also discussed. Finally, a summary is given together with suggestions for future research.

Keywords: master physician scheduling problem, preferences, bi-objective optimization, mathematical programming.

1 Introduction

Personnel scheduling is defined as the process of constructing optimized work schedules for staff (Topaloglu, 2009). A literature review of applications, models and algorithms in personnel scheduling has been provided by Ernst et al. (2004). The personnel scheduling problem includes a wide variety of applications such as airlines, railways, manufacturing and health care systems. In this paper, the scheduling of physicians in a hospital is addressed.

Brandeau et al. (2004) provided a more recent collection of Operations Research applications in health care, with particular emphasis on health care delivery. To our knowledge, research on physician scheduling focuses primarily on a single type of duty, such as the emergency room (e.g. Vassilacopoulos, 1985; Beaulieu et al., 2000; Carter and Lapierre, 2001; Gendreau et al., 2007; Puente, et al., 2009), the operating room (e.g. Testi et al., 2007; Burke and Riise, 2008; Beliën et al., 2009; Roland et al., 2009), the physiotherapy and rehabilitation services (Ogulata et al., 2008).

In this paper, our problem, termed the **Master Physician Scheduling Problem**, is the tactical planning problem of assigning physician activities to the time slots over a time horizon incorporating a large number of rostering and resource constraints together with complex

Aldy Gunawan
School of Information Systems, Singapore Management University
80 Stamford Road, Singapore 178902
E-mail: aldygunawan@smu.edu.sg

Hoong Chuin Lau
School of Information Systems, Singapore Management University
80 Stamford Road, Singapore 178902
E-mail: hclau@smu.edu.sg

physician preferences. The main objectives are to satisfy as many physicians’ preferences and duty requirements as possible while ensuring optimum usage of available resources such as clinics and operating theatres.

The major contributions/highlights of this paper are as follows:

(1) We take a physician-centric approach to solving this problem, since physician retention is the most critical issue faced by hospital administrations worldwide.

(2) We formulate the problem as a bi-objective optimization problem and solve the problem by different methods: ε–Constraint Method, Weighted-Sum Method and Hill-Climbing Algorithm.

The organization of the paper is as follows. Section 2 gives some literature review. Section 3 gives a detailed description of the master physician scheduling problem. In Section 4, we propose a bi-objective mathematical programming model along with the description of notation and variables, constraints and objective functions. Section 5 discusses three different methods used to solve the problem. Section 6 makes a computational analysis of the model with a real case from the Surgery Department of a large local government hospital, as well as on randomly generated problem instances. Finally, we provide some concluding perspectives and directions for future research in Section 7.

2 Literature Review

There have been a number of review papers in the area of personnel scheduling and rostering research, as in the works of Aggarwal (1982), Burke et al. (2004), Ernst et al. (2004). Much of the research on personnel scheduling in health care has been devoted to the case of nurse scheduling problem (e.g. Burke et al., 2004; Ernst et al, 2004; Bard and Purnomo, 2005; Beliën and Demeulemeester, 2005; Petrovic and Berghe, 2008). On the other hand, little work has been done on the physician scheduling problem. Carter and Lapierre (2001) provide the fundamental differences between physicians and nurses scheduling problems. Unlike nurse rostering problems, in physician scheduling, maximizing satisfaction only matters, as physician retention is the most critical issue faced by hospital administrations. In addition, while nurse schedules must adhere to collective union agreements or written rules, physician schedules are more driven by personal preferences and with no formal scheduling rules.

Physician and nurse scheduling problems are typically multi-objective by nature. One approach for handling multi-objective optimization problem is to formulate the objectives as soft constraints and define the global objective function as the total deviations in the soft constraints (Beaulieu, et al., 2000; Topaloglu, 2006, 2009; Burke et al., 2009). Another way to solve a multi-objective problem is to apply the Weighted-Sum method that combines the objectives into a single scalar value (Beaulieu et al., 2000, Carter and Lapierre, 2001; Blöchliger, 2004; Topaloglu, 2006; Beliën et al., 2009; Puente et al., 2009; Topaloglu, 2009). Yet another method that has also been considered is the sequential method (Topaloglu, 2009). In this method, objectives are sorted in descending order of importance and optimized in an iterative procedure. Another most commonly used method is goal programming since it allows simultaneous solution of multiple objectives (Ozkarahan, 2000; Ogulata and Erol, 2003; Topaloglu, 2006; White et al., 2006).

Burke et al. (2007) and Burke et al. (2009) presented a Pareto-based optimization technique based on a Simulated Annealing algorithm to address nurse scheduling problems in the real world. One of the latest papers about physician rostering problem is presented by Puente et al. (2009). The problem consists in designing timetables for the physicians at the Emergency Department in a hospital.

3 Problem Definition

This paper focuses on a physician scheduling problem for the Surgery Department of a large government hospital in Singapore. The problem (termed the Master Physician Scheduling Problem) is to assign different physician duties (or activities) to the defined time slots over a time horizon incorporating a large number of constraints and complex physician preferences. For simplicity, we assume the time horizon to be one work week (Mon-Fri), further partitioned into 5 days and 2 shifts (AM and PM).

The work mode combines shifts and duties. Physicians may specify their respective *ideal schedule* in terms of the duties they like to perform on their preferred days and shifts, as well as shifts-off or days off. An actual schedule is generated by taking the physicians' preferences together with resource capacity and rostering constraints into consideration (Figure 1).

Due to conflicting constraints, the ideal schedules might not be fully satisfied in the actual schedule (see Figure 1 for illustration). That may occur in two possible scenarios:

- Some duties have to be scheduled on different shifts or days – which we term *non-ideal scheduled duties* (e.g. Physician 2 Tuesday duties).
- Some duties simply cannot be scheduled due to resource constraints – which we term *unscheduled duties* (e.g. Physician 1 Friday PM duty).

		Physician 1	Physician 2	…	Physician $\|I\|$
Monday	AM	Duty 1	-	…	Duty 3
	PM	Duty 5	Duty 4	…	Duty 1
Tuesday	AM	-	Duty 1	…	Duty 5
	PM	Duty $\|L\|$	Duty 5	…	Duty 2
	:	:	:		:
	:	:	:		:
Friday	AM	Duty 4	-	…	Duty $\|L\|$
	PM	Duty 1	Duty $\|L\|$	…	-

Physicians' Ideal Schedule

		Physician 1	Physician 2	…	Physician $\|I\|$
Monday	AM	Duty 1	-	…	Duty 5
	PM	Duty 5	Duty 4	…	Duty 1
Tuesday	AM	-	Duty 5	…	Duty 3
	PM	Duty $\|L\|$	Duty 1	…	-
	:	:	:		:
	:	:	:		:
Friday	AM	Duty 4	-	…	Duty $\|L\|$
	PM	-	Duty $\|L\|$	…	-

Actual Schedule

Figure 1. Example of the master physician scheduling problem

The master physician scheduling problem is a highly constrained resource allocation problem. The constraints imposed are categorizes into two different types: hard and soft constraints. Our goal is to meet the hard constraints while aiming at a high-quality result with respect to soft constraints. The hard constraints in our problem are as follows:

- *H*1: No physician can perform more than one duty in any shift.
- *H*2: The number of resources (e.g. operating theatres, clinics) needed cannot exceed their respective capacities at any time. For simplicity, we assume that each type of activity does not share its resources with another type of activities – for example, operating theatres and clinics are used to perform surgery and out-patient duties, respectively.
- *H*3: The number of activities allocated to each physician cannot exceed his contractual commitments, and do not conflict with his external commitments. In this paper, we assume external commitments take the form of physicians' request for shifts-off or days-off, and hence no duty should be assigned to these requests.

The master physician scheduling problem incorporates both physician preferences and ergonomic constraints, optimizing on two objectives - maximizing the number of ideal schedules and minimizing the number of unscheduled duties. These objectives are related to the following soft constraints:

- *S*1: Duties should be scheduled with respect to the *ideal schedule*.
- *S*2: For some heavy duties, such as surgery and endoscopy duties, that could not be scheduled with respect to the ideal schedule, we try to reschedule these duties with respect to the ergonomic constraints:
  - If a physician is assigned to a heavy duty in the morning shift, then he cannot be assigned to another type of heavy duty in the afternoon shift *on the same day*. However, it is possible to assign *the same* type of heavy duties in consecutive shifts on the same day.
  - Similarly, a physician cannot also be assigned to another type of heavy duty in the morning shift on a particular day if he has been assigned to a heavy duty in the afternoon shift on the previous day.

4 Mathematical Programming Model

The following notation is required to formulate the mathematical programming model.

Parameters

I = Set of physicians, $i \in \{1,2,\ldots,|I|\}$

J = Set of days, $j \in \{1,2,\ldots,|J|\}$

K = Set of shifts per day, $k \in \{1,2,\ldots,|K|\}$

L = Set of duties, $l \in \{1,2,\ldots,|L|\}$

L^H = $\{ l \in L : l =$ *heavy* duty$\}$

PRA = $\{ (i,j,k) \in I \times J \times K : (i,j,k)$ = physician i requests not being assigned on day j shift $k\}$

R_l = number of resources required to perform duty l $(l \in L)$

C_{jkl} = number of resources available for duty l on day j shift k $(j \in J, k \in K, l \in L)$

(i.e. resource capacity)

A_{il} = number of duty l requested by physician i in a weekly schedule $(i \in I, l \in L)$

F_{ijkl} = 1 if physician i requests duty l on day j shift k (*ideal schedule*), 0 otherwise $(i \in I, j \in J, k \in K, l \in L)$

Decision and auxiliary variables

X_{ijkl} = 1 if physician i is assigned to duty l on day j shift k with respect to the ideal schedule, 0 otherwise

Y_{ijkl} = 1 if physician i is assigned to duty l on day j shift k with respect to the ergonomic constraints, 0 otherwise

U_i = number of unscheduled duties of physician i

N_i = number of non-ideal scheduled duties of physician i

S_i = number of ideal scheduled duties of physician i

We consider the problem that optimizes physician ideal schedules on one hand, and on the other, improves the quality of duty transition on non-ideal scheduled slots through ergonomic constraints. More precisely, we are concerned with the bi-objective problem of maximizing the number of ideal scheduled duties (1) and minimizing the number of unscheduled duties under ergonomic constraints (2).

Maximize $Z_1 = \sum_{i \in I} S_i$ (1)

Minimize $Z_2 = \sum_{i \in I} U_i$ (2)

subject to:

$R_l \times \sum_{i \in I}(X_{ijkl} + Y_{ijkl}) \le C_{jkl}$ $\quad j \in J, k \in K, l \in L$ (3)

$X_{ijkl} + Y_{ijkl} \le 1$ $\quad i \in I, j \in J, k \in K, l \in L$ (4)

$\sum_{j \in J} \sum_{k \in K}(X_{ijkl} + Y_{ijkl}) \le A_{il}$ $\quad i \in I, l \in L$ (5)

$\sum_{l \in L}(X_{ijkl} + Y_{ijkl}) \le 1$ $\quad i \in I, j \in J, k \in K$ (6)

$\sum_{l \in L}(X_{ijkl} + Y_{ijkl}) = 0$ $\quad (i, j, k) \in PRA$ (7)

$X_{ijkl} \le F_{ijkl}$ $\quad i \in I, j \in J, k \in K, l \in L$ (8)

$U_i = \sum_{l \in L} A_{il} - \sum_{j \in J} \sum_{k \in K} \sum_{l \in L}(X_{ijkl} + Y_{ijkl})$ $\quad i \in I$ (9)

$S_i = \sum_{j \in J} \sum_{k \in K} \sum_{l \in L} X_{ijkl}$ $\quad i \in I$ (10)

$N_i = \sum_{j \in J} \sum_{k \in K} \sum_{l \in L} Y_{ijkl}$ $\quad i \in I$ (11)

$Y_{ijkl_1} + X_{ij(k+1)l_2} + Y_{ij(k+1)l_2} \le 1$ $\quad i \in I, j \in J, k \in \{1,2,\ldots,|K|-1\}, l_1 \,\&\, l_2 \in L^H (l_1 \ne l_2)$ (12)

$Y_{ij|K|l_1} + X_{i(j+1)1l_2} + Y_{i(j+1)1l_2} \le 1$ $\quad i \in I, j \in \{1,2,\ldots,|J|-1\}, l_1 \,\&\, l_2 \in L^H (l_1 \ne l_2)$ (13)

$$Y_{ij|K|l_1} + X_{ij1l_2} + Y_{ij1l_2} \le 1 \qquad i \in I, j \in J, l_1 \,\&\, l_2 \in L^H (l_1 \ne l_2) \tag{14}$$

$$Y_{ij1l_1} + X_{i(j-1)|K|l_2} + Y_{i(j-1)|K|l_2} \le 1 \qquad i \in I, j \in \{2,3,\ldots,|J|\}, l_1 \,\&\, l_2 \in L^H (l_1 \ne l_2) \tag{15}$$

$$X_{ijkl}, Y_{ijkl} \in \{0,1\} \qquad i \in I, j \in J, k \in K, l \in L \tag{16}$$

$$U_i, N_i, S_i \in Z^+ \qquad i \in I \tag{17}$$

Constraint (3) ensures that the total number of resources required does not exceed total number of available resources per shift (the resource capacity constraint). Note that R_l is set to zero for activities without limited number of resources available. (4) ensures that a duty is scheduled as either an ideal or a non-ideal duty. (5) represents the number of duties allocated to each physician cannot exceed his contractual commitments. (6) ensures that each physician cannot be assigned more than one duty in any shift, while (7) ensures that no duty would be assigned to a physician during any shifts-off or days-off requested. Duties represented by X_{ijkl} have to be scheduled with respect to the *ideal schedule* (constraint (8)). Constraints (9), (10) and (11) calculate the number of unscheduled duties, ideal scheduled duties and non-ideal scheduled duties, respectively. The details of ergonomic constraints are represented by (12) – (16). Finally, (16) imposes the 0-1 restrictions for the decision variables X_{ijkl} and Y_{ijkl} while (17) is the nonnegative integrality constraint for the decision variables U_i, N_i and S_i.

In the following section, three different approaches are proposed to solve the bi-objective physician scheduling problem: one based on ε–Constraint approach that obtains a single solution, and the others based on Weighted-Sum Method and Hill-Climbing Algorithm that obtains non-dominated or Pareto-optimal solutions.

5 Proposed Methods

5.1 ε–Constraint Method

The ε–Constraint Method was suggested by Haimes et al. (1971). In this method, the bi-objective problem is reformulated by just keeping one of the objective functions and restricting the other objective function within user-specified value. Here, we decide to restrict the number of unscheduled duties to be less than or equal to the values obtained by solving another model proposed by Gunawan and Lau (2009) (denote by U_i^*). Therefore, the model only focused on minimizing the number of unscheduled duties with respect to ergonomic constraints. The modified problem is as follows:

[**ε–Constraint Model**]

Maximize $Z_1 = \sum_{i \in I} S_i$ (18)

subject to:

constraints (3) – (17)

$$U_i \leq U_i^* \qquad i \in I \qquad (19)$$

5.2 Weighted-Sum Method

The Weighted-Sum Method is the simplest approach and commonly used to solve the multiple-objective optimization problem. It formulates the problem as a classical multi-objective weighted-sum model that combines two objectives into a single objective by multiplying each objective with a user-defined weight. The weight of each objective is usually chosen in proportion to the objective's relative importance in the problem.

[**Weighted-Sum Model**]

Minimize $Z = W_1 \times \left(-\sum_{i \in I} S_i\right) + W_2 \times \left(\sum_{i \in I} U_i\right)$ (20)

subject to: constraints (3) – (17)

Note that in Weighted-Sum Model, the original objective function Z_1 is transformed into a minimization objective function. The advantage of the Weighted-Sum method is that it guarantees finding Pareto-optimal solutions for convex optimization problems, which can be inferred from Deb (2003) Theorem 3.1.1:

Corollary: The solution to the Weighted-Sum Model is not Pareto-optimal iff either W_1 or W_2 is set to zero.

Algorithm

(1) Set $W_1 = 1$
(2) **Repeat**
(3) Set $W_2 = 1 - W_1$
(4) Solve the Weighted-Sum Model optimally (using mathematical programming)
(5) $W_1 = W_1 - 0.1$
(6) **Until** $W_1 < 0$
(7) For all solutions generated by the above, let M denote the subset of Pareto-optimal solutions
(8) **For** a pre-set number of iterations **do** the following
(9) Let M_1 and M_2 ($\in M$) with the lowest and the second lowest total number of unscheduled duties, respectively
(10) Set $W'_1 = W_1$ of solution M_1 and $W'_2 = W_2$ of solution M_1
(11) Set $W''_1 = W_1$ of solution M_2 and $W''_2 = W_2$ of solution M_2
(12) Calculate new weight values, denoted as W^*_1 and W^*_2, as follows:
$W^*_1 = (W'_1 + W''_1)/2$
$W^*_2 = 1 - W^*_1$
(13) Solve the Weighted-Sum Model with $W_1 = W^*_1$ and $W_2 = W^*_2$
(14) **If** the solution obtained is a new Pareto-optimal solution
(15) **Then** update M
(16) **Else if** the solution obtained and M_1 are the same
(17) Set the solution obtained as M_1 and Update M
(18) **Else if** the solution obtained and M_2 are the same
(19) Set the solution obtained as M_2 and Update M

Figure 2. Algorithm to obtain Pareto-optimal solutions

In this paper, instead of using a single set of weight values, several different sets of weight values would be used to efficiently generate a set of Pareto-optimal solutions. First, a constant k

number of solutions with different values of W_1 uniformly distributed between [0, 1] are generated. Since not all Pareto-optimal solutions may be discovered by the initial set of weight values, we introduce an adaptive exploration on the neighborhood of weight values using linear interpolation, i.e. we examine two different Pareto-optimal solutions to derive weight values for obtaining other possible optimal solutions. The detail of the algorithm is presented in Figure 2.

5.3 Hill-Climbing Algorithm

In this section, we turn to a Hill Climbing Algorithm to generate a set of non-dominated solutions. The initial solution is generated by setting one of the weight values to 1. Next, a set M of *potentially non-dominated solutions* would be generated. This set is updated whenever a new non-dominated solution $\mathbf{x}'$ is generated. This updating process consists of two possible actions:

(1) Adding $\mathbf{x}'$ to M if there is no other solution $\mathbf{v} \in M$ such that $\mathbf{v}$ dominates $\mathbf{x}'$

(2) Removing all solutions from set M which are dominated by $\mathbf{x}'$

The Hill-Climbing Algorithm will terminate when either there is no unscheduled duties or it reaches a pre-set number of iterations. The algorithm is given as follows.

Hill-Climbing Algorithm
(1) Generate a starting solution $\mathbf{x} \in D$, where D is the set of feasible solutions
(2) $M := \varnothing$
(3) Update M of *potentially efficient solutions* with $\mathbf{x}$
(4) Repeat
(5) Select one solution $\mathbf{x} \in M$
(6) Construct a new solution $\mathbf{x}' \in V(\mathbf{x})$, where $V(\mathbf{x}) \subseteq D$ is the neighborhood of solution $\mathbf{x}$
(7) **If** a new solution $\mathbf{x}'$ exists
(8) **If** $\mathbf{x}'$ is a non-dominated solution then
(9) Update M
(10) **Until** the stop conditions are satisfied

Figure 3. Hill-Climbing Algorithm

Our proposed neighborhood structure is in essence a kind of ejection chain move involving either one or two physicians and the pool of hitherto unscheduled duties. From the initial solution generated, the *Unscheduled_Pool* contains the list of physicians with the respective number of unscheduled duties. A physician (say physician *i*) and one of his unscheduled duty (say *Duty*1) is selected randomly from the *Unscheduled_Pool* and the aim is to insert it into the schedule, thereby decreasing the total number of unscheduled duties by 1. To do so, one of his scheduled duties (say *Duty*2) at say *slot*2 needs to be reallocated to another timeslot say *slot*1.

Note that each time as a duty is moved to another timeslot, it must satisfy either one of the two following conditions:

Condition1: the duty is allocated to a timeslot that follows the physician's ideal schedule. The net effect is that the total number of ideal scheduled duties either remains the same or increases by 1.

Condition2: the duty is allocated to a timeslot that does *not* follow the physician ideal schedule. In this case, we need to ensure that the ergonomic constraint is not violated. The net effect is that the total number of ideal scheduled duties either remains the same or decreases by 1.

In considering the relocation of *Duty*1 to *slot*2, two possible scenarios are possible:

(1) **Scenario 1**: If there is resource available at *slot*2 to perform *Duty*1 (Figure 4), the move can be performed.

(2) **Scenario 2**: If no resource is available *slot*2 for *Duty*1 (Figure 5), then another physician *j*, who is performing the *same* duty (i.e. *Duty*1) at *slot*2 will be selected (if any) and we apply an ejection chain strategy to swap out the *Duty*1 of physician *j* so as to free up the resource needed.

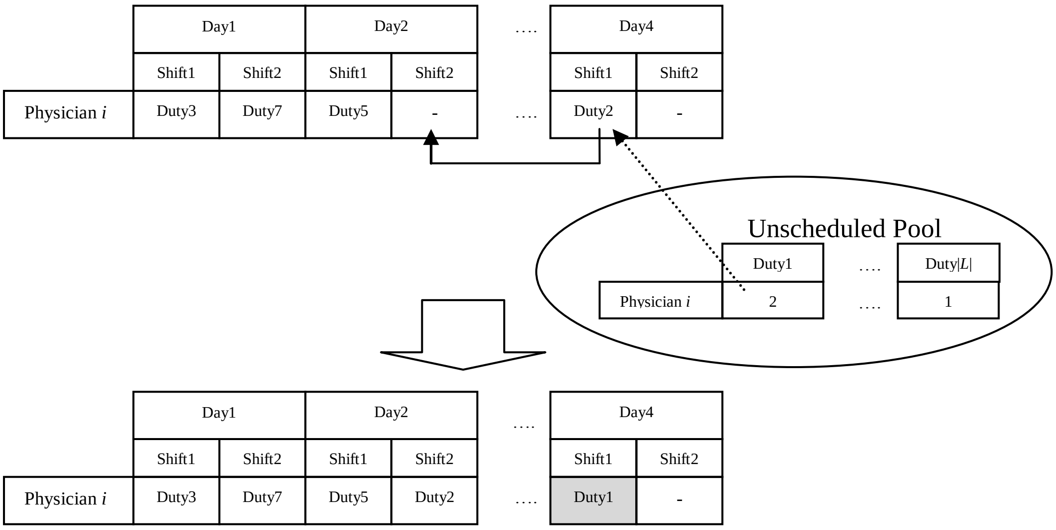

Figure 4. Illustration of Scenario 1

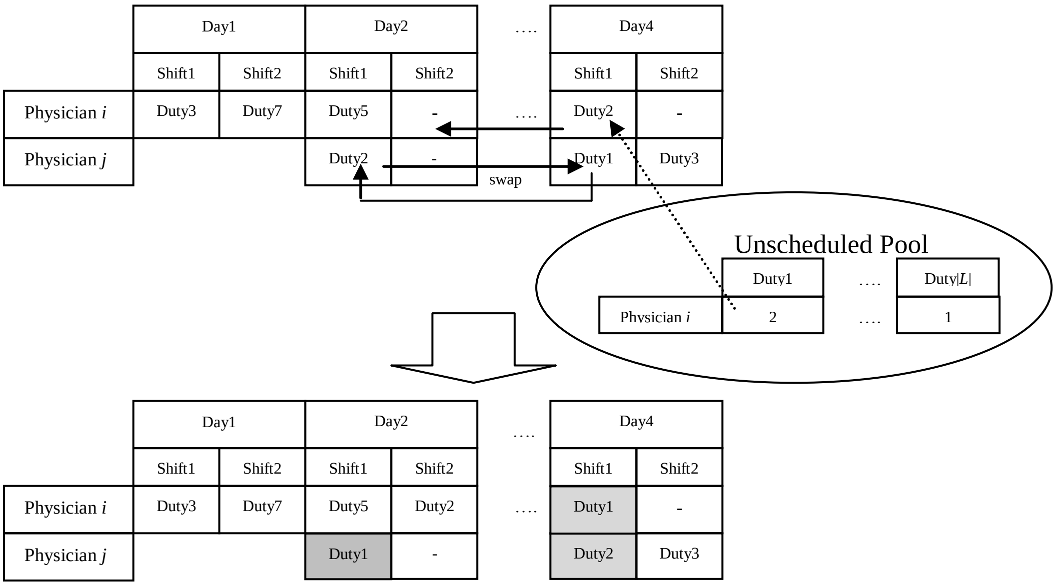

Figure 5. Illustration of Scenario 2

The pseudo-code for generating moves for this neighborhood is as shown in Figure 6.

(1) Select physician *i* from *Unscheduled_Pool* randomly
(2) Find an empty timeslot randomly, *slot*1
(3) By considering all scheduled duties of physician *i*, find one possible time *slot*2 such that the duty allocated at *slot*2 can be reassigned to *slot*1
(4) **If** it can be rescheduled at *slot*1,
(5) Find an unscheduled duty of physician *i*, *Duty*1
(6) **If** the resource capacity at *slot*2 for *Duty*1 is greater than 0
(7) Evaluate whether *Duty*1 can be allocated to *slot*2
(8) If there is no constraint violation, generate a new possible solution $\mathbf{x}'$
(9) **Else if** the resource capacity at *slot*2 for *Duty*1 is equal to 0
(10) Evaluate whether *Duty*1 can be allocated to *slot*2
(11) **If** there is no conflict,
(12) Find a physician *j* who has the same duty scheduled, *Duty*1, at *slot*2
(13) Apply an ejection chain strategy to physician *j*, by ensuring that all constraints are satisfied
(14) **If** there is no constraint violation, generate a new possible solution $\mathbf{x}'$

Figure 6. Neighborhood Move

6 Computational Results

To evaluate the performance of the proposed methods, computational experiments were done on 6 different random problem sets and a real case from the Surgery Department of a large local government hospital. The 6 sets problem sets were generated with varying values of the parameter – total percentage of *heavy* duties assigned to physicians (last column of Table 1). For each problem set, we also generate several problem instances with different values of number of resources available in every shift (Table 2). The details about how problem instances were generated are summarized in Gunawan and Lau (2009).

Table 1. Characteristics of problem instances

Problem Set	Number of physicians	Number of shifts per day	Number of days	Number of duties	Number of *heavy* duties	Number of duties with limited capacity	Total percentage of *heavy* duties*
Case study	15	2	5	9	3	3	73%
Random 1	20	2	5	7	3	3	20%
Random 2	20	2	5	7	3	3	30%
Random 3	20	2	5	7	3	3	40%
Random 4	20	2	5	7	3	3	50%
Random 5	20	2	5	7	3	3	60%
Random 6	20	2	5	7	3	3	70%

$* = \left(\sum_{i\in I}\sum_{l\in L^H} A_{il} \,/\, |I|\times|J|\times|K|\right)\times 100\%$

In the following sub-sections, we report a suite of computational results and analysis obtained from the proposed methods. The mathematical programming models (ε–Constraint and Weighted-Sum Models) were implemented using ILOG OPL Studio 5.5 and the proposed algorithm (Hill Climbing Algorithm) was coded in C++. All codes are executed on a Intel (R) Core (TM)^2 Duo CPU 2.33GHz with 1.96GB RAM that runs Microsoft Windows XP.

Table 2. Examples of varying values of C_{jkl} (Random 1 and Random 2 instances)

Problem Set	Instances	L		
		Duty 1	Duty 2	Duty 3
Random 1		15	28	22
	Random 1a	3	6	4
	Random 1b	3	5	4
	Random 1c	3	4	4
	Random 1d	3	3	4
	Random 1e	3	3	3
	Random 1f	2	3	3
	Random 1g	1	2	2
Random 2		21	46	32
	Random 2a	4	10	5
	Random 2b	4	9	5
	Random 2c	4	8	5
	Random 2d	4	7	5
	Random 2e	4	6	5
	Random 2f	4	5	5
	Random 2g	4	5	4
	Random 2h	3	5	4
	Random 2i	2	4	3

6.1 Results from ε–Constraint Method

As described in Section 5.1, the physician scheduling problem is reformulated by keeping one objective and restricting the other one within a specified value. In this paper, we restrict the number of unscheduled duties within the number of unscheduled duties generated by another model proposed by Gunawan and Lau (2009).

In Gunawan and Lau (2009), the ergonomic constraints are imposed to all scheduled duties. On the other hand, in this paper, duties are scheduled with respect to either of two criteria: the number of scheduled duties with respect to the physicians' ideal schedules has to be satisfied as many as possible, while non-ideal scheduled duties cannot violate ergonomic constraints.

Table 3. Computational results of ε–Constraint Model

Problem Instances	Number of unscheduled duties	Number of scheduled duties		Percentage of unscheduled duties	Percentage of scheduled duties	
		Ideal	Non-ideal		Ideal	Non-ideal
Case study	8	135	7	5.3	90.0	4.7
Random 1a	0	196	4	0.0	98.0	2.0
Random 1b	0	192	8	0.0	96.0	4.0
Random 1c	0	192	8	0.0	96.0	4.0
Random 1d	4	186	10	2.0	93.0	5.0
Random 1e	5	181	14	2.5	90.5	7.0
Random 1f	5	180	15	2.5	90.0	7.5
Random 1g	10	173	17	5.0	86.5	8.5
Random 2a	0	196	4	0.0	98.0	2.0
Random 2b	0	196	4	0.0	98.0	2.0
Random 2c	0	196	4	0.0	98.0	2.0
Random 2d	0	194	6	0.0	97.0	3.0
Random 2e	0	194	6	0.0	97.0	3.0
Random 2f	3	186	11	1.5	93.0	5.5
Random 2g	3	186	11	1.5	93.0	5.5
Random 2h	3	183	14	1.5	91.5	7.0
Random 2i	10	174	16	5.0	87.0	8.0

Table 3 summarizes the results obtained for the real case study, as well as Random 1 and 2 instances. In general, we found that the number of unscheduled duties is relative small compared with the number of ideal scheduled duties (less than or equal to 5.3%). By using this method, different optimal solutions can be found by setting different U_i^* values. Take note however that it is possible that infeasible solutions would be obtained.

The following table summarizes the average percentages of all our problem sets. It can be observed that the average percentage of ideal scheduled duties is at least 86%, and only Random 5 has the average percentage of non-ideal scheduled duties which is more than 10%.

Table 4. Summary of computational results of ε–Constraint Model

Problem Set	Number of instances	Average percentage of unscheduled duties	Average percentage of scheduled duties	
			Ideal	Non-ideal
Case study	1	5.3	90.0	4.7
Random 1	7	1.7	92.9	5.4
Random 2	9	1.1	94.7	4.2
Random 3	9	1.8	91.2	6.9
Random 4	11	1.1	89.6	9.2
Random 5	13	1.1	86.7	12.2
Random 6	15	2.6	89.5	7.9

6.2 Results from Weighted-Sum Method

In Section 5.2, we proposed an algorithm to generate several possible sets of weight values in order to obtain set of Pareto-optimal solutions. It is started by generating 10 different sets of weight values that uniformly distributed within [0, 1].

In the next step, we set the number of iterations to 5 iterations. This step is applied for further finding of other possible Pareto-optimal solutions. By using linear interpolation, we focus on exploring neighborhoods of the solutions with the lowest values of the total number of unscheduled duties since we view that unscheduled duties as bad compared to non-ideal scheduled duties.

In general, the value of W_1 should be less than 0.5 in order to obtain the lowest number of unscheduled duties. We also found that the higher the percentage of heavy duties, the lower the value of W_1 should be set. It could be due to the difficulty to assign unscheduled heavy duties with respect to ergonomic constraints. That’s why we need to give higher importance/value for W_2.

Table 5 represents the results obtained by the proposed algorithm. Here, we only present two representative instances 1g and 6l for illustration purposes. Figure 7 represents the Pareto-optimal solutions obtained for Random 1 and 2 instances.

In general, we observe that the more we increase the weight value of the first objective (W_1), the less we get the number of non-ideal scheduled duties (see Table 5 for illustration). At the same time, the number of unscheduled duties would also be increased since the number of unscheduled duties becomes less important with the decreased weight value of the second objective (W_2). This method could guarantee finding solutions on the Pareto-optimal set. However, we also found that different weight values need not necessarily lead to Pareto-optimal solutions and some sets of weight values might lead to the same solution.

Table 5. Computational results of instances 1g and 6l

Random 1g					Random 6l				
Weight		Number of Scheduled duties		Number of Unscheduled duties	Weight		Number of Scheduled duties		Number of Unscheduled duties
W_1	W_2	Ideal	Non-Ideal		W_1	W_2	Ideal	Non-Ideal	
1.0	0.0	183	0	17	1.0	0.0	181	0	19
0.9	0.1	183	3	14	0.9	0.1	181	9	10
0.8	0.2	183	3	14	0.8	0.2	181	9	10
0.7	0.3	183	3	14	0.7	0.3	181	9	10
0.6	0.4	183	3	14	0.6	0.4	181	9	10
0.5	0.5	181	7	12	0.5	0.5	181	9	10
0.4	0.6	179	11	10	0.4	0.6	179	13	8
0.3	0.7	179	11	10	0.3	0.7	173	22	5
0.2	0.8	179	11	10	0.2	0.8	169	27	4
0.1	0.9	179	11	10	0.1	0.9	160	38	2
0.0*	1.0	31	51	139	0.0*	1.0	113	85	2
0.45	0.55	179	11	10	0.15	0.85	160	38	2
0.475	0.525	179	11	10	0.175	0.825	165	32	3
0.4875	0.5125	179	11	10	0.1625	0.8375	160	38	2
0.49375	0.50625	179	11	10	016875	0.83125	165	32	3
0.496875	0.503125	179	11	10	0.165625	0.834375	160	38	2

* Non Pareto-optimal solution

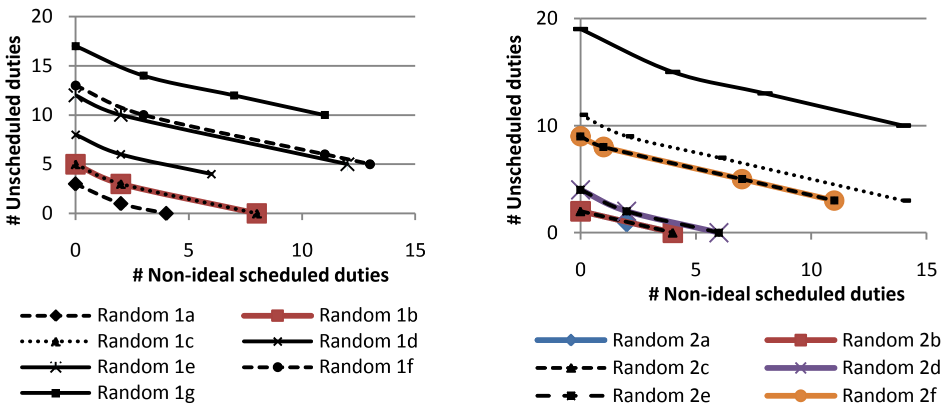

Figure 7. Pareto-optimal solutions of Random 1 and 2 problem sets

The proposed algorithm is also tested to the real case study (Table 6).The value of W_1 should be within [0.9, 1.0] in order to obtain the lowest number of unscheduled duties. The result of the real case study problem by the ε–Constraint and the Weighted-Sum Methods and the actual allocation generated manually by the hospital are also compared.

The number of ideal scheduled duties obtained by the Weighted-Sum Model is significantly higher than that of the manual allocation. Although the number of unscheduled duties obtained by both ε–Constraint Model and Weighted-Sum Model are slightly worse than the number of unscheduled duties via manual allocation, the number of non-ideal scheduled duties is better than that of the manual allocation. One of possible reason is in the manual allocation, the administrator allocates non-ideal scheduled duties to any time slots/shifts without considering the ergonomic constraints. In the manual allocation, there are also two physicians who have to cancel their days-off or shifts-off for other duties. This outcome is very undesirable since they might have external commitments that cannot be delayed or cancelled.

Table 6. Comparison between the manual allocation and model solutions on a real case

	Manual allocation	ε–Constraint Model	Weighted-Sum Model
Number of unscheduled duties	5	8	8
Number of non-ideal scheduled duties	10	7	2
Number of ideal scheduled duties	135	135	140

6.3 Results from Hill-Climbing Algorithm

In this experiment, the number of iterations for Hill Climbing is set to 200 for each test instance. Note that the number of Pareto-optimal solutions obtained by the Weighted-Sum Method is small. For instance, for problem instances Random 1 (i.e. 1a to 1g), the number of Pareto-optimal solutions generated is between 3 and 4, compared with the Hill-Climbing Algorithm which provides up to 10 non-dominated solutions (see Table 7). Figure 8 represents results obtained by the Hill-Climbing Algorithm for some of the representative instances.

Table 7. The number of solutions generated

Problem Set	The range of the number of solutions generated	
	Weighted-Sum Method	Hill-Climbing Algorithm
Case Study	1	2
Random 1	[3,4]	[3,10]
Random 2	[2,4]	[3,11]
Random 3	[3,5]	[2,12]
Random 4	[4,5]	[4,12]
Random 5	[4,5]	[4,16]
Random 6	[4,7]	[4,11]

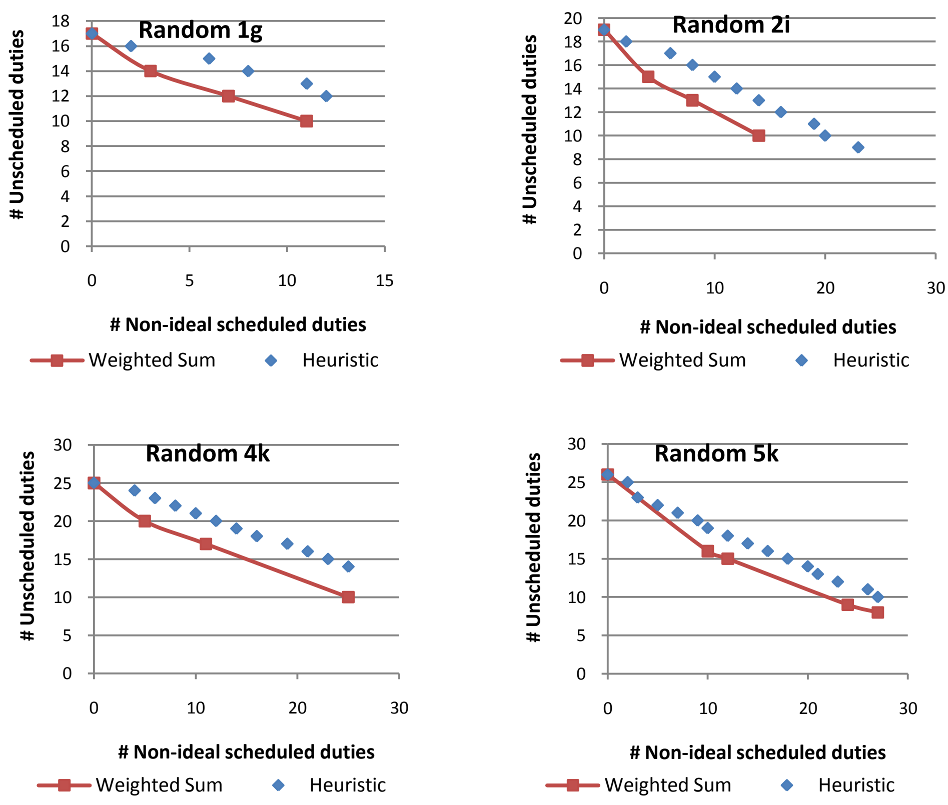

Figure 8. Non-dominated solutions of Hill-Climbing Method

As observed by Burke et al., 2009, one issue in comparing the algorithms in multi-objective problems is that there is no systematic criterion to measure the performance of each algorithm. In Burke et al. (2009), a number of objective functions were considered, and their approach was convert these objectives into goals, and the aim was to minimize the deviations (i.e. the percentage of total number of violations in the solution with respect to the total number of constraints).

In this paper, we choose to measure the deviation of our heuristic approach from Pareto optimality directly. Let the results obtained by the Hill-Climbing and Weighted-Sum Methods be denoted as Sets H and W with sizes of n_H and n_W, respectively. In order to compare and measure the closeness between a solution $\mathbf{x} \in H$ and a solution $\mathbf{y} \in W$, we propose the following formula:

$$dist(\mathbf{x},\mathbf{y}) = \left[\frac{|Z_1(\mathbf{y}) - Z_1(\mathbf{x})|}{Z_1(\mathbf{y})} + \frac{|Z_2(\mathbf{y}) - Z_2(\mathbf{x})|}{Z_2(\mathbf{y})} \right] \Big/ 2 \tag{21}$$

For a particular solution $\mathbf{x}$, we calculate n_W different values of $dist(\mathbf{x},\mathbf{y})$ and choose the solution $\mathbf{y}$ which yields the minimum $dist(\mathbf{x},\mathbf{y})$ value (ties broken arbitrarily). The fitness value of a solution $\mathbf{x}$ is calculated as follows:

$$Fitness(\mathbf{x}) = \frac{1}{dist(\mathbf{x},\mathbf{y}) + 1} \tag{22}$$

Note that this is a normalized value that falls between 0 and 1, where a value 1 means perfect fit, and tends to 0 as the distance increases.

For each problem instance, we will have n_H different values of $dist(\mathbf{x},\mathbf{y})$. For example, for Random 1g (see Figure 8), six different non-dominated solutions were obtained by the Hill-Climbing Algorithm. The average fitness value associated with a given problem instance is then calculated as follows:

$$Average\ Fitness = \frac{\sum_{\mathbf{x} \in H} Fitness(\mathbf{x})}{n_H} \tag{23}$$

Table 8 lists the distances obtained for representative instances Random 1g, 3i, 4k and 5k. We observe that the Hill-Climbing Algorithm produces non-dominated solutions with the fitness values greater than 0.93. Although the results obtained by the Hill-Climbing Method might not be Pareto-optimal solutions, we found that the number of non-dominated solutions generated is more than that of the Weighted-Sum Method. For future research, these non-dominated solutions can be considered as starting points/initial solutions that would be further improved in order to obtain Pareto-optimal solutions.

Table 8. Comparison between the Hill-Climbing Algorithm and the Weighted-Sum Method

Problem Instances	Number of solutions generated by Weighted-Sum Method	Number of solutions generated by Hill Climbing Algorithm	Average Fitness
Random 1g	4	6	0.974
Random 2i	4	11	0.962
Random 4k	4	12	0.954
Random 5k	5	16	0.937

Table 9 summarizes the statistical descriptions of the entire results for all problem sets. The grand mean of average fitness values is above 0.9 which is considered high. Some instances in Random 2 and 3 have the values of 1. The **Grand Mean** column refers to the means of the average fitness values of the respective problem sets.

Table 9. Summary of average fitness values

Problem Set	Number of instances	Grand Mean	Std dev	Minimum	Maximum
Case Study	1	0.96	0.04	0.93	0.99
Random 1	7	0.94	0.03	0.90	0.97
Random 2	9	0.95	0.03	0.92	1.00
Random 3	9	0.96	0.03	0.92	1.00
Random 4	11	0.94	0.01	0.92	0.96
Random 5	13	0.96	0.02	0.93	1.00
Random 6	15	0.94	0.02	0.92	0.97

7 Conclusion

In this paper, we introduce the master physician scheduling problem considering two different objectives simultaneously. Three different multi-objective methods have been proposed. These approaches were tested on a real case from the Surgery Department of a large local government hospital, as well as on randomly generated problem instances. We observe that the objectives were better satisfied compared against the manual allocation.

In terms of future research, there are several potential areas for investigation. An interesting research direction would be to apply other methods, such as Multi-Objective Simulated Annealing, Multi-Objective Tabu Search, and to develop other neighborhood structures in an attempt to improve the solutions. In the same way, we can also consider other constraints, such as fairness constraints, which commonly seen in other hospitals (Gendreau et al., 2007). Another systematic criterion to measure the performance of an algorithm can be considered as future work. We notice that some distance values of the Hill-Climbing Method's solutions might be large. It is probably due to the limitation of the Weighted-Sum Method in generating all possible Pareto-optimal solutions. The application of the ε–Constraint Method is rather limited in this paper; for example, we can consider applying this method to retrieve the complete Pareto-optimal solutions. The main idea is to construct a sequence of ε-Constraint Model based on a progressive modification of U_i^* values (equation (19)) (Deb, 2003; Bérubé et al., 2009).

Acknowledgements We like to thank the Department of Surgery, Tan Tock Seng Hospital (Singapore) for providing valuable comments and test situations.

References

1. Aggarwal, S. (1982). A focused review of scheduling in services. *European Journal of Operational Research*, *9*(*2*), 114-121.
2. Bard, J. F., & Purnomo, H. W. (2005). Preference scheduling for nurses using column generation. *European Journal of Operational Research, 164,* 510-534.

3. Beaulieu, H., Ferland, J. A., Gendron, B., & Philippe, M. (2000). A mathematical programming approach for scheduling physicians in the emergency room. *Health Care Management Science*, *3*, 193-200.
4. Beliën, J., & Demeulemeester, E. (2005). Integrating nurse and surgery scheduling. In Proceedings of the 2nd Multidisciplinary International Scheduling Conference: Theory and Applications 2005, New York, USA, 18-21 July 2005, 408-409.
5. Beliën, J., Demeulemeester, E., & Cardoen, B. (2009). A decision support system for cyclic master surgery scheduling with multiple objectives. *Journal of Scheduling*, *12*, 147-161.
6. Bérubé, J.-F, Gendreau, M., & Potvin, J.-Y. (2009). An exact ε-constraint method for bi-objective combinatorial optimization problems: application to the traveling salesman problem with profits. *European Journal of Operational Research*, *194*, 39-50.
7. Blöchliger, I. (2004). Modeling staff scheduling problems. A tutorial. *European Journal of Operational Research*, *158*, 533-542.
8. Brandeau, M.L., Sainfort, F., & Pierskalla, W.P. (2004). *Operations research and healthcare: A handbook of methods and applications*. Dordrecht: Kluwer Academic.
9. Burke, E. K., & Riise, A. (2008). Surgery allocation and scheduling. In Proceedings of the 7th International Conference of Practice and Theory of Automated Timetabling 2008, Montreal, Canada, 18-22 August 2008.
10. Burke, E. K., De Causmaecker, P., Vanden Berghe, G., & Van Landeghem H. (2004). The state of the art of nurse rostering. *Journal of Scheduling*, *7*(*6*), 441-499.
11. Burke, E. K., Li, J., & Qu, R. (2009). A Pareto-based search methodology for multi-objective nurse scheduling. *Annals of Operations Research*, DOI: 10.1007/s10479-009-0590-8, published online.
12. Burke, E. K., Li, J., Petrovic, S. & Qu, R. (2007). A new Pareto-optimality based metaheuristic approach to the multi-objective nurse scheduling problem. Technical Report, School of Computer Science and IT, University of Nottingham.
13. Carter, M. W., & Lapierre, S. D. (2001). Scheduling emergency room physicians. *Health Care Management Science*, *4*, 347-360.
14. Deb, K. (2003). Multi-objective optimization using evolutionary algorithms. Wiley & Sons, Chichester, New York.
15. Ernst, A. T., Jiang, H., Krishnamoorthy, M., Owens, B., & Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, *153*, 3-27.
16. Gendreau, M., Ferland, J., Gendron, B., Hail, N., Jaumard, B., Lapierre, S., Pesant, G., & Soriano, P. (2007). Physician scheduling in emergency rooms. In Burke, E. K., & Rudová, H. (eds.), The Practice and Theory of Automated Timetabling VI (PATAT'06, Selected papers). *Lecture Notes in Computer Science*, *3867*, 53-66, Springer, Heidelberg.
17. Gunawan, A., & Lau, H.C. (2009). Master physician scheduling problem. In Proceedings of the 4th Multidisciplinary International Scheduling Conference 2009, Dublin, Ireland, 10 – 12 August 2009.

18. Haimes, Y.Y., Lasdon, L.S., & Wismer, D.A. (1971). On a bicriteriona formulation of the problems of integrated system identification and system optimization, *IEEE Trans. Syst., Man Cybernet*, *1*, 296-297.
19. Ogulata, S. N., & Erol, R. (2003). A hierarchical multiple criteria mathematical programming approach for scheduling general surgery operations in large hospitals. *Journal of Medical Systems*, *27*(*3*), 259-270.
20. Ogulata, S. N., Koyuncu, M., & Karakas, E. (2008). Personnel and patient scheduling in the high demanded hospital services: a case study in the physiotherapy service. *Journal of Medical Systems*, *32*(*3*), 221-228.
21. Ozkarahan, I. (2000). Allocation of surgeries to operating rooms by goal programming. *Journal of Medical Systems*, *24*(*6*), 339-378.
22. Petrovic, S., & Vanden Berghe, G. (2008). Comparison of algorithms for nurse rostering problems. In Proceedings of the 7th International Conference of Practice and Theory of Automated Timetabling 2008, Montreal, Canada, 18 – 22 August 2008.
23. Puente, J., Gómez, A., Fernández, I., Priore, P. (2009). Medical doctor rostering problem in a hospital emergency department by means of genetic algorithm. *Computers & Industrial Engineering*, *56*, 1232-1242.
24. Roland, B., Di Martinelly, C., Riane, F. & Pochet, Y. (2009). Scheduling an operating theatre under human resource constraints. *Computers & Industrial Engineering*, article in press, corrected proof, available online, 14 January 2009.
25. Testi, A., Tanfani, E., & Torre, G. (2007). A three-phase approach for operating theatre schedules. *Health Care Management Science*, *10*, 163-172.
26. Topaloglu, S. (2006). A multi-objective programming model for scheduling emergency medicine residents. *Computers & Industrial Engineering*, *51*, 375-388.
27. Topaloglu, S. (2009). A shift scheduling model for employees with different seniority levels and an application in healthcare. *European Journal of Operational Research*, *198*, 943-957.
28. Vassilacopoulos, G. (1985). Allocating doctors to shifts in an accident and emergency department. *Journal of the Operational Research Society*, *36*, 517-523.
29. White, C.A., Nano, E., Nguyen-Ngoc, D.H, White, & G.M. (2006). An evaluation of certain heuristic optimization algorithms in scheduling medical doctors and medical students. In Proceedings of the 6th International Conference of Practice and Theory of Automated Timetabling 2006, Brno, The Czech Republic, 30 August – 1 September 2006.