Universiti Malaysia Pahang examination timetabling problem: scheduling invigilators

MN Mohmad Kahar[1,2*] and G Kendall[1,3]

[1]University of Nottingham, Nottingham, UK; [2]Universiti Malaysia Pahang, Kuantan, Pahang Malaysia; and [3]University of Nottingham, Malaysia Campus

This paper presents a real-world examination timetabling problem from Universiti Malaysia Pahang (UMP), Malaysia. The problem involves assigning invigilators to examination rooms. This problem has received less attention than the examination timetabling problem from the research community partly because no data sets are available in the literature. In modelling, and solving, this problem we assume that there is already an examination timetable in place (this was the subject of our previous work) and the task is to assign invigilators to that timetable. The contributions of this paper are to formally define the invigilator scheduling problem and to present a constructive algorithm that is able to produce good quality solutions that are superior to the solutions produced when using the university's current software. We also include additional constraints taking into account the comments made by the invigilators, which the current system fails to capture. The model we present, we believe, accurately reflects the real-world problem, capturing various aspects of the problem that have not been presented before in the scientific literature. Moreover, the proposed approach adheres to all hard constraints, which the university's current system fails to do.

Journal of the Operational Research Society (2014) **65,** 214–226. doi:10.1057/jors.2012.6
Published online 13 March 2013

Keywords: optimisation; timetabling; invigilator; scheduling

1. Introduction

Examination timetabling problems are an important task, performed by educational institutions across the world. The process (if carried out manually) is often time consuming and difficult. It may even be difficult to produce a feasible solution, let alone a solution that satisfies all those who it affects (Laporte and Desroches, 1984; Carter and Laporte, 1996; McCollum *et al*, 2009). Numerous definitions of the examination timetabling problem can be found in the literature, which can be summarised as assingning exams to a specific or limited number of timeslots and rooms, with the aim of satisfying both the hard constraints (eg, a conflict free timetable) and the soft constraints as much as possible (eg, spread student exams evenly). Hard constraints cannot be broken and a timetable is considered feasible if all the hard constraints are satisfied. An example of a hard constraint is that no student should be required to sit two examinations simultaneously. Soft constraints (or the objective) are requirements that are not essential but should be satisfied as far as possible, and the level of violation is used to evaluate the quality of the solution. An example of a soft constraint is to evenly spread exams throughout the exam period.

Many papers discussing the examination timetable problem can be found in the literature. The PATAT conference series of selected papers (ie, Burke and Ross, 1996; Burke and Carter, 1998; Burke and Erben, 2001; Burke and De Causmaecker, 2003; Burke and Trick, 2005; Burke and Rudova, 2007) are a particularly good reference. However, besides the problem of scheduling exams to timeslots and/or rooms, the educational examination timetabling problem does not end there. The problem also involves assigning invigilators to the exam/room. This is normally done after the institution has generated the *exam-timeslot-room* timetable (Burke *et al*, 1996a).

Most of the research found in the literature involves assigning exams to timeslots and/or rooms. Only a few papers have investigated invigilator scheduling (Burke *et al*, 1996a; Reis and Oliveira, 1999; Cowling *et al*, 2003; Ong *et al*, 2009). One reason for invigilation scheduling receiving less attention from the research community is due to the fact that no data sets are available.

In our view, there are three ways an instituition could implement invigilator assignment; by hiring outside staff, using their own staff or by using a mixture of in-house staff and outside staff. This point is further discussed in Section 3.

Correspondence: MN Mohmad Kahar, FSKKP, Universiti Malaysia Pahang, Lebuhraya Tun Razak, 26300, Gambang, Kuantan, Pahang, Malaysia.
E-mail: mnizam@ump.edu.my

Invigilator scheduling contains many hard and soft constraints, which vary greatly from one institution to another. An example of a hard constraint is that invigilators are not assigned to multiple invigilation duties at the same time. A typical soft constraint specifies that invigilation duties need to be evenly spread among the invigilators. In a survey (Burke *et al*, 1996a), it was found that 29% of universities agree that the task of invigilator scheduling is a major problem. This is also reported by Cowling *et al* (2003) and Ong *et al* (2009) where many invigilators are not satisfied with their individual schedule.

This paper investigates the invigilator scheduling problem taken from Universiti Malaysia Pahang (UMP). This invigilator data set contains numerous constraints, which we believe have never been discussed or modelled before.

In Section 2, we concisely describe the examination timetabling problem, particularly the *exam-timeslot-room* assignment. We present related work on invigilator scheduling in Section 3. A description of the UMP examination invigilation problem, including the constraints, is discussed in Section 4. A formal model of the problem is presented in Section 5. In Section 6, we describe the experimental setup for our proposed strategy. Section 7 gives a description of the data set used in our experiments. In Section 8, a comparison between the solutions achieved between the current method employed by UMP (which is produced using proprietary software) and our method is presented in order to evaluate the effectiveness of the proposed methodology. We discuss the additional invigilator constraints and the results in Sections 9 and 10, respectively. In Sections 11 and 12, we summarise our contribution and present our conclusions together with possible future research directions.

2. Examination timetabling

According to Reis and Oliveira (1999), the exam timetabling problem can be defined as:

> ET—examination timetabling: Scheduling (in time) of the exams of a set of university courses avoiding overlapping exams having common students and spreading the exams for the students as much as possible. Room assignment and invigilator assignment can be done prior to or after the exam timetabling phase.

Based on this definition, the entire examination timetabling problem involves exams, timeslots, rooms and invigilators. However, most of the research found in the scientific literature investigates the *exam-timeslot-room* assignment problem, concentrating on the algorithmic performance with the aim of producing good quality solutions in minimal time (see Qu *et al*, 2009). The scheduling of invigilators is often ignored. The most common examination timetabling data sets found in the literature are the Toronto data set (Carter *et al*, 1996), the Nottingham data set (Burke *et al*, 1996b) and the Melbourne data set (Merlot *et al*, 2003). Many papers, which use this data set, can be found in the PATAT conference series of selected papers and Qu *et al* (2009). The Toronto, Nottingham and Melbourne data sets only cover one-third of the examination timetabling problem, as their focus is on assigning exams to timeslots (although the Nottingham and Melbourne data sets do consider maximum seating capacity in a timeslot). More recently, the second International Timetabling Competition (ITC2007) data set has been introduced (McCollum *et al*, 2010), which includes more realistic problems than the benchmark data sets. Although the ITC2007 data set contains many constraints and, more closely resembles the real-world problem, it is still lacking with respect to invigilator scheduling that forms part of the complete educational examination timetabling problem (Burke *et al*, 1996a; Hussin, 2005).

Currently, there is no recognised data set for the invigilator scheduling problem in the scientific literature (as far as the authors are aware). In our opinion, invigilator scheduling has been largely overlooked by the scientific community, despite being as important as *the exam-timeslot-room* assignment problem to the institution. As well as being reported in Burke *et al* (1996a), it is also reported by Cowling *et al* (2003) and Ong *et al* (2009) that many invigilators are dissatisfied with their individual schedule and would prefer a better schedule. Therefore, this scheduling problem seems to be worthy of investigation.

3. Invigilator scheduling problem

The invigilator schedule affects many people, not least of all the invigilators themselves. In our view, invigilator scheduling varies widely from one institution to another and it is not possible to provide a common definition. The invigilator scheduling problem can be defined as assigning invigilators (ie, staff and/or non-staff) to exams and/or rooms in such a way that the hard constraints are not violated and the soft constraints are satisfied as far as possible. Examples of these constraints are presented in Cowling *et al* (2003), Hussin (2005), Awang *et al* (2006) and Burke *et al* (1996a):

- Invigilator duties must be assigned fairly (ie, equivalent duties for all invigilators).
- Invigilators are not assigned to more than one room in a timeslot.
- Invigilators do not invigilate their own papers.
- Invigilators should not exceed a maximum number of invigilation duties.

In our view, invigilator scheduling can be divided into three categories with respect to the staff that are employed to carry out the invigilations.

(I1) *Outside staff*: The institution hires non-staff (typically these are from outside the institution) to invigilate the exam timetable. This approach reduces the complexity of the problem, as we only need to consider fulfilling the requested number of invigilators for each exam/room.

(I2) *In-house staff*: The institution use their own staff to invigilate the exams (Ong *et al*, 2009). Some insitutions use only academic staff (eg, lecturers), while others might also include non-academic staff (eg, administrators, technicians, postdoctoral researchers, etc). The academic staff are often assigned as chief invigilators, while non-academic staff are assigned to help in the invigilation process. Compared with (I1), this approach may have a significant number of constraints such as invigilators not being able to invigilate their own exam paper (or alternatively being expected to), not being assigned to more than one invigilation duty at a time, the invigilation duties being evenly spread among the staff, etc.

(I3) *Mixed*: The institution use their own staff and hire outside staff to invigilate the exam timetable. The mixing of staff types provides flexibility to the institution, as it enables a variety of working methods to be adopted (eg, in-house staff act as chief invigilators and outside staff to provide the relevant support).

The implementation of I1 would increase operational costs, as the institution needs to pay for the outside staff. In our opinion, a mix of outside and in-house staff (I3) gives more advantages and flexibility to the timetabling office compared with I1 and I2. However, it also comes at an increase in operational cost. It also reduces the complexity of the problem compared with I2. However, we recognise that every institution operates in different ways and the staffing model that is adopted is dependent on many factors and what is suitable for one institution may not be suitable for another.

UMP only uses its own staff as invigilators (I2). This results in numerous constraints such as the chief invigilators must be a member of academic staff, staff are required to carry out a number of invigilations within the exam period, etc. A detailed description of the UMP invigilator constraints is presented in Section 4.

An exam timetable is often generated by first assigning exams to timeslots (*exam-timeslot* assignment). A further process then assigns rooms and/or invigilators after the exam timetable has been approved/accepted (Burke *et al*, 1996a). It is evident in the literature that most published work only deals with *exam-timeslot* assignment. Only a few papers have addressed the *exam-classroom* assignment (Dammak *et al*, 2006) and very little work can be found on invigilator scheduling. A lecturer preference survey by Cowling *et al* (2003) reveals that:

- Invigilators prefer two to three invigilation duties with a 1- or 2-day gap between each duty.
- Lecturers with other responsibilities (eg, administrative or research work) should be given a reduced number of invigilation duties.
- An adequate gap is given between invigilation duties and the lecturers' own papers. This is to allow the lecturers enough time to do their marking and submit their grades within the required time.
- A fair distribution of chief invigilator duties.

Ong *et al* (2009) developed an invigilation scheduling system concentrating on optimising lecturer preferences (ie, invigilation dates, time and constraints) for UiTM Sarawak (Samarahan Campus). The invigilation scheduling only involves lecturers (I2). Previously, the schedule was prepared manually by the institution's invigilation scheduling committee. They randomly assigned invigilation duties and, later, there was a lot of swapping among the lecturers. This resulted in confusion, misunderstanding and complaints of uneven invigilation duty distribution. This motivated them to develop an invigilation system with the aim of optimising lecturer preferences. The system enables lecturers to view the examination timetable, choose their preferred invigilation timeslots, specify the examination date and the time of their own subjects, and view their individual schedule and the final exam/invigilation timetable. Reis and Oliveira (1999) experimented with an examination timetabling problem from the University Fernando Pessoa, Porto using constraint logic programming. They solve the problem by scheduling each exam into an available timeslot. For each exam, one or several exam rooms are allocated and for each room, a set of invigilators is defined. The proposed approach included the following investigations:

- Scheduling exams into timeslots and, once completed, scheduling the rooms. Finally, they deal with invigilator scheduling.
- Schedule exams into rooms, then schedule the timeslot and then the invigilators.
- Schedule exams into timeslots, then schedule rooms and invigilators simultaneously.
- The exams, timeslots, rooms and invigilators are scheduled simultaneously.

A survey carried out by Awang *et al* (2006) on the UMP examination timetable asked about invigilator satisfaction with their invigilation timetable. It revealed that most of the invigilators are not satisfied with the gap between invigilation duties and the number of invigilations. They suggested that each invigilation duty should have at least a 2- or 3-day gap. However, they prefer fewer invigilation duties, considering that they also need to be available/on standby during their own exam paper. They requested an even spread of invigilation duties among the staff. As the timetable officer is open to any suggestions for improving the current timetable, we are motivated to include the suggestion above as an additional constraint in addition to the original constraints. These additional constraints are discussed further in Section 9.

In this paper, we solve the UMP examination timetable in two phases: first, we schedule the exams into timeslot and rooms simultaneously (Kahar and Kendall, 2010). We then use the solution from the first phase as input to the invigilator scheduling phase. The scheduling of exams into timeslots, rooms and last the invigilators has been reported as the best sequence in order to produce a good quality solution (Reis and Oliveira, 1999). Our proposed approach to this second phase is presented in Section 5, but first we describe the problem informally, and then present a formal definition.

4. UMP: invigilator scheduling

The UMP Academic Office is responsible for planning and managing all academic processes. This is done with the aid of an Information Management System (IMS), which is proprietary software. The IMS generates the examination timetable, assigns exams to timeslot, allocates rooms and schedules invigilators. However, the IMS also involves manual processes in order to achieve a feasible solution and it is unable to determine the quality of the solutions it produces due to having no underlying mathematical model (that we are aware of) that allows the quality of the generated timetable to be measured. This paper focuses on scheduling invigilators onto the exam timetable. Therefore, one of our research objectives is to develop a formal model for the invigilator scheduling problem in order to evaluate the quality of the solution produced by the IMS. We can then use the same model to measure the quality of the solutions we produce.

The UMP invigilator scheduling problem description includes the following:

- A complete examination timetable is already available.
- List of staff: A set of available staff to be scheduled into rooms.
- Staff status: Whether staff are academic or otherwise as only certain staff can carry out certain duties.
- Number of invigilators required in each room.
- List of exam paper(s) that each lecturer teaches.
- Hard constraints that must be satisfied.
- Soft constraints that contribute to a penalty if they are violated.

The hard constraints for the UMP invigilator scheduling problem are as follows:

(H1) Invigilators or chief invigilators cannot invigilate their own exam paper. This is because they need to be on standby during their exam paper to assist students with any queries (see Equation (8)).

(H2) With the extra tasks and responsibility associated with a chief invigilator, university policy only allow staff with lecturer status to act as a chief invigilator (see Equation (9)).

(H3) Staff are not assigned to more than one invigilation duty in one timeslot (see Equation (10)).

(H4) Staff can only invigilate a maximum of three examinations within the exam period (see Equation (11)).

(H5) Each room should be assigned the required number of invigilators—including a chief invigilator (see Equation (12)).

In measuring the quality of the solution, the soft constraints are as follows:

(S1) The chief invigilator duties should be evenly spread among the lecturers (see Equation (2)).

(S2) The invigilation duties (invigilator and chief invigilator) should be evenly spread among all staff (see Equation (5)).

5. Problem formulation

Indices

i,j	$1 \ldots N$, where N is the number of examinations.
l	$1 \ldots L$, where L is the number of staff.
r	$1 \ldots R$, where R is the number of rooms.
t	$1 \ldots T$, where T is the number of timeslots.

Parameters

S_l	The status of staff l. 1 denotes a lecturer, 0 otherwise.
l_r	The number of invigilators (including chief invigilator) required in each room r .
a_{il}	The exam-staff matrix where each element (denoted by a_{il}, $i\in\{1 \ldots N\}$ and $l\in\{1 \ldots L\}$) denoted as 1 corresponding as the staff teaches the course (or exam paper) in that semester, 0 otherwise.

Examination timetabling parameters
Note: These variables are set by the examination scheduling phase (see Kahar and Kendall, 2010).

x_{it}	1 if examination i is assigned to timeslot t, 0 otherwise.

y_{ir} 1 if examination i is assigned to room r, 0 otherwise.

z_{rt} 1 if room r is assigned to timeslot t, 0 otherwise.

Decision variables

v_{lrt} 1 if staff l is assigned to invigilate in room r in timeslot t as an invigilator, 0 otherwise.

w_{lrt} 1 if staff l is assigned to invigilate in room r in timeslot t as the chief invigilator, 0 otherwise.

The objective function is as follows:

$$\text{Minimise, } F(x) = F_1 + F_2 \tag{1}$$

The first component of the objective function, F_1, is that the chief invigilator duties should be evenly spread among the lecturers in the staff list L ($S_l = 1$).

$$F_1 = \sum_{l=1}^{L} \textit{Chief duty}(w_{lrt}) \tag{2}$$

where

$$\text{Chief duty } (w_{lir}) = \begin{cases} 0 & \text{if } \sum\limits_{t=1}^{T} \sum\limits_{r=1}^{R} w_{lrt} \leqslant \lceil CId \rceil \\ 20 & \text{otherwise} \end{cases} \tag{3}$$

The maximum number of chief invigilation duties assigned to every lecturer ($S_l = 1$) can be calculated based on the number of lecturers in the staff list L and the number of rooms selected in the *exam-timeslot-room timetable* solution. The number of chief invigilation duties is calculated by taking the ceiling value of *CId*. The calculation is as follows:

$$\lceil CId \rceil = \frac{\sum_{t=1}^{T} \sum_{r=1}^{R} z_{rt}}{\sum_{l=1}^{L} S_l} \tag{4}$$

The second component of the objective function F_2 is concerned with the even spread of both invigilator and chief invigilator duties.

$$F_2 = \sum_{l=1}^{L} \text{staff duty}(v_{lir}, w_{lir}) \tag{5}$$

where

$$\text{staff duty } (v_{lir}, w_{lir}) = \begin{cases} 0 & \text{if } \sum\limits_{t=1}^{T} \sum\limits_{r=1}^{R} (v_{lrt} + w_{lrt}) \leqslant \lceil Id \rceil \\ 20 & \text{otherwise} \end{cases} \tag{6}$$

The maximum number of invigilation duties for all staff L can be calculated based on the number of invigilators (l_r) required in each room (from the *exam-timeslot-room timetable* solution) and the number of staff L. The required number of invigilation duties for each member of staff is calculated by taking the ceiling value of *Id*. The calculation is as follows:

$$\lceil Id \rceil = \frac{\sum_{t=1}^{T} \sum_{r=1}^{R} z_{rt} l_r}{L} \tag{7}$$

The objective function (Equation (1)) is subject to the following constraints:

(a) Invigilators cannot invigilate their own exam paper (H1).

$$\sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{r=1}^{R} (a_{il} x_{it} y_{ir})(v_{lrt} + w_{lrt}) = 0 \quad \text{For all } l \in \{1, \ldots, L\} \tag{8}$$

(b) The chief invigilators must be a lecturer, $S_l = 1$ (H2).

$$w_{lrt} \leqslant S_l \quad \text{For all } l \in \{1, \ldots, L\},\ t \in \{1, \ldots, T\} \text{ and } r \in \{1, \ldots, R\} \tag{9}$$

(c) Staff are not assigned to more than one invigilation duty at a time (H3).

$$\sum_{r=1}^{R} (v_{lrt} + w_{lrt}) \leqslant 1 \quad \text{For all } l \in \{1, \ldots, L\}, \text{ and } t \in \{1, \ldots, T\} \tag{10}$$

(d) All staff are required to invigilate a maximum of three examinations within the exam period (H4).

$$\sum_{t=1}^{T} \sum_{r=1}^{R} (v_{lrt} + w_{lrt}) \leqslant 3 \quad \text{For all } l \in \{1, \ldots, L\}. \tag{11}$$

(e) The total number of invigilators (including one as chief invigilator) assigned to each room r in timeslot t has to equal the number of invigilators required for each room l_r (H5).

$$\sum_{l=1}^{L} (v_{lrt} + 2w_{lrt}) = z_{rt}(l_r + 1) \quad \text{For all } r \in \{1, \ldots, R\} \text{ and } t \in \{1, \ldots, T\} \tag{12}$$

6. Experimental setup

In this section, we present our proposed invigilator scheduling algorithm in order to solve the UMP problem. As described previously, invigilator scheduling is a post process from the *exam-timeslot-room timetable* process (Kahar and Kendall, 2010). Therefore, the information (eg, rooms, exams, timeslot, etc) from the *exam-timeslot-room* assignment phase is already known and, hence the results that produce the minimum cost value are retained from this first phase. Even if several runs were made in the first phase, the run that produced the minimum cost value is saved.

As described in Section 5, the chief invigilator assignment is the most critical part as it involves the most constraints: must be a lecturer, cannot invigilate their own paper, etc. Invigilator assignment is less complicated as the member of staff can be a lecturer, or otherwise.

Algorithm parameters:

- $l = 1 \ldots L$, where L is the number of staff available for the invigilation duties.
- $r = 1 \ldots roomSelected$, where *roomSelect* is a list of selected rooms in the *exam-timeslot-room* assignment solution.
- $m = 1 \ldots l_r$, *where* l_r is the number of invigilators required in room r.
- $c = 1 \ldots C$, where C is the number of candidates list.
- S_l status of staff (ie, lecturer or other) l. 1 denoted as a lecturer, 0 otherwise.
- D_l holds the total invigilation duty for staff l.
- *totalCostValue*[*c*] store the cost value for assigning invigilator l to timeslot and room in candidates list c.
- $v_{lrt} = 1$ if staff l is assigned to invigilate in room r in timeslot t as an invigilator, 0 otherwise.
- $w_{lrt} = 1$ if staff l is assigned to invigilate in room r in timeslot t as the chief invigilator, 0 otherwise.

Hence, we have designed an algorithm that first concentrates on assigning the chief invigilators to all the rooms, followed by other invigilator assignments.

The algorithm (see Figure 1) starts (Line 2) by sorting staff L in ascending order based on D_l or randomly. Next in Line 3, we calculate the ceiling invigilation value for chief, *ceilingCId* (Equation (4)) and invigilator duties, *ceilingID* (Equation (7)). Then, we assign a chief invigilator into room in the *roomSelected* list (Step 2, Line 4). The first staff in L is selected. The number of chief invigilator we consider is referred to as candidates list (which we use during the random ordering strategies) and we show the effect of different candidates list sizes in the result section. If l is a lecturer ($S_l = 1$) and satisfies the following: l does not teach the exam (H1), has no other invigilation duty within the same timeslot (H3) and does not exceed the maximum number of chief invigilation duties (H4), we then calculate the penalty value on assigning the selected invigilator to r and store the information in *totalCost Value*[*c*] (Lines 10–14). We also consider the invigilator should be in the same timeslot and building as their own exam if on duty during their exam constraint (H6) in this step during the additional constraints experiments. Next, increase c to search of other l for the candidates list. Then, increase l, however, if l is greater than L, we set $l = 1$ and assign $totalCostValue[c] = +\infty$ (which means that there are no available invigilator in *totalCostValue*[*c*])

```
1   Step 1: set-up
2      Sort Staff L in ascending order based on D_l or randomly
3      Calculate the ceiling value ceilingCId (eq.4) and ceilingId (eq.7)
4   Step 2: Assign chief invigilators to room
5      Set r ← 1
6      Until r = roomSelected repeat:
7      (2.1) Set l ← 1
8      (2.2) Set c ← 1
9      (2.3) Until c = C, repeat:
10         (2.3.1) If l ≤ L and l is a academic staff (S_l = 1), then calculate the cost value F and store in totalCostValue[c], simultaneously increase c
11                   s.t. l does not teach the exam (H1), no other invigilation duty within the same timeslot(H3), does not exceed the maximum
12                      invigilation duty (H4) and Invigilator on duty during their exam must be on the same building (H6) - optional
13         (2.3.2) Increase l
14         (2.3.3) If l > L, set l ← 1, totalCostValue[c] ← +∞
15     (2.4) Select the minimum total cost value from C, set W_lrt←1 and update D_l if totalCostValue[c]≠ +∞  for every c
16     (2.5) Increase r
17     (2.6) Sort Staff L in ascending order based on D_l or randomly
18  Step 3: Assign invigilators to room
19     Set r ← 1
20     Until r = roomSelected repeat:
21     (3.1) Set l ← 1
22     (3.2) Set m ← 1
23     (3.3) Until m = l_r-1 repeat:
24         (3.3.1) Set c ← 1
25         (3.3.2) Until c = C, repeat:
26             (3.3.2.1) If l ≤ L, then calculate the cost value F and store in totalCostValue[c], simultaneously increase c
27                         s.t. l does not teach the exam (H1), no other invigilation duty within the same timeslot(H3), does not exceed the maximum
28                            invigilation duty (H4) and Invigilator on duty during their exam must be on the same building (H6) - optional
29             (3.3.2.2) Increase l
30             (3.3.2.3) If l > L, set l ← 1, totalCostValue[c] ← +∞
31         (3.3.3) Select the minimum total cost value from C, set V_lrt←1 and update D_l if totalCostValue[c]≠ +∞  for every c
32         (3.3.4) Increase m
33     (3.4) Increase r
34     (3.5) Sort Staff L in ascending order based on D_l or randomly
35   Step 4: Verification and Cost value
36     Verify the solution and Calculate Cost Value (Eq.1)
```

Figure 1 Pseudocode for the *invigilator scheduling*.

(Lines 13–14). The search continues by selecting the minimum total cost value in C (ie, *totalCostValue*[C]) and set the corresponding l into the selected timeslot and room, $W_{lrt}=1$ and subsequently increase D_l (Lines 15). Finally, we increase r (Line 16) and sort staff L in ascending order based on D_l (this would let the search to always select the minimum number of invigilation duties of staff L) or randomly (Line 17).

Next, we assign the invigilators (Step 3, Line 18). The same process is carried out as for assigning chief invigilators except now, the search will continue for a l_r-1 of duration for each *roomselected* (Line 23). l_r is the number of invigilators required in *roomselected.* For example, if $l_r=4$, then the search will iterate three times (which is equivalent to three invigilators and one chief invigilator). Lastly, the algorithm verifies whether the solution complies with all the hard constraints and calculates the cost of the solution (Line 36).

7. UMP invigilator data set

Experiments were carried out with two different data sets from semester1-2007/2008 and semester1-2008/2009. The data are obtained from the solution generated by the UMP proprietary software. We noticed that there is a difference in the information (ie, staff status, number of lecturers, etc) provided by the Academic Office compared with the actual solution that they provided us with. Therefore, we decided to use the data from the schedule that was actually used as this more accurately represents what was done in practice. A description of the data sets is given below.

In semester1-2007/2008, the number of staff available for invigilation duties is 227. Of those, 152 are lecturers and 75 are non-lecturers. Each room must be allocated two invigilators (including the chief invigilator). 169 lecturers are involved in teaching the 157 exams. The 169 lecturers are not all necessarily included in the staff list, L. In semester1-2008/2009, the number of staff available for invigilation duty is 332. Of those, 207 are lecturers and 125 are non-lecturers. The total number of invigilators required by each room varies from a minimum of two to a maximum of four (including the chief invigilator). 194 lecturers are involved in teaching the 165 exams. The 194 lecturers are not all necessarily included in the staff list, L.

8. Results

In this section, we present the results of the invigilator timetable generated by the UMP proprietary software by inputting their solution into the model described in Section 5. A comparison of the result obtained by the UMP proprietary software with our proposed algorithm (Section 6) is also discussed. The results are summarised in Table 1.

8.1. Semester1-2007/2008

Analysing the solution produced by the UMP proprietary software in the *exam-timeslot-room* assignment phase, a total of 269 rooms were used. Therefore, using these 269 rooms the invigilator scheduling problem exhibits the following characteristics (see Table 1, Column A).
Hard constraints: From the constraints in Section 5, the invigilator timetable produced by UMP only complies with two out of the five hard constraints violating the following:

(i) Constraint (H1): Staff are assigned to invigilate their own exam paper. Supposedly, they need to be available during the exam of their own paper to answer any queries.
(ii) Constraint (H4): Staff are assigned to more than three exams, which exceeds the maximum number of invigilation duties within the exam period.
(iii) Constraint (H5): One room was not assigned the required number of invigilators.

Soft constraints: The objective of the invigilator scheduling solution is measured based on two objectives. The cost value for F_1 (Equation (2)) is 220 and F_2 (Equation (5)) is 20 with a total cost value of 240.

8.2. Semester1-2008/2009

Based on the result produced by the UMP proprietary software, 290 rooms have been used. The invigilator scheduling solution for semester 1 2008/2009 exhibits the following characteristics (see Table 1, Column A).

Hard constraints: The invigilator scheduling produced by UMP violates all five of the hard constraints listed in Section 5.
Soft constraints: The cost value of the invigilator timetable solution for F_1 (Equation (2)) is 20 and F_2 (Equation (5)) is 120 with a total cost value of 140.

8.3. Proposed solution approach

In scheduling invigilators, our experiments use the *exam-timeslot-room* solution produced by the UMP proprietary software for semester1-2007/2008 and semester1-2008/2009 (see Table 1, Column B). We also use a solution from our own approach based on a graph colouring heuristic approach (Kahar and Kendall, 2010; see Table 1, Column C). The experiments were run on a Pentium core2 processor. The average running time was ≈ 23 s. However, the running time depends on the number of rooms being selected in the *exam-timeslot-room* assignment phase. Obviously, a higher number of rooms would slightly increase the running time, but this is not of particular significance.

Table 1 Invigilator scheduling results using additional constraint as described in Section 5

Constraints	*(A)* *Proprietary software*		*(B)* *Our approach using exam timetable from UMP*				*(C)* *Our approach using exam timetable from Kahar and Kendall* (2010)			
	*Semester*1-2007/2008 *(269 rooms)*	*Semester*1-2008/2009 *(290 rooms)*	*Semester*1-2007/2008 *(269 rooms)*		*Semester*1-2008/2009 *(290 rooms)*		*Semester*1-2007/2008 *(244 rooms)*		*Semester*1-2008/2009 *(274 rooms)*	
			*c*1 $\approx 23\,s$	*c*5 $\approx 28\,s$	*c*1 $\approx 52\,s$	*c*5 $\approx 62\,s$	*c*1 $\approx 22\,s$	*c*5 $\approx 26\,s$	*c*1 $\approx 53\,s$	*c*5 $\approx 60\,s$
(H1) Invigilators or chief invigilators cannot invigilate their own exam paper	Not (1)	Not (2)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
(H2) Only allow staff with lecturer status to act as a chief invigilator	Yes	Not (1)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
(H3) Staff are not assigned to more than one invigilation duty in one timeslot	Yes	Not (2)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
(H4) Staff can only invigilate a maximum of three examinations within the exam period	Not (1)	Not (6)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
(H5) Each room should be assigned the required number of invigilators (including a chief invigilator)	Not (1)	Not (2)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Cost value functions ($F=F_1+F_2$)	240	140	0	0	0	0	0	0	0	0

*c*1 = candidates list of one; *c*5 = candidates list of five; Not (*x*) = not comply (number of violations); Yes = comply.

Using least invigilation duties ordering strategies on the UMP solution from semester1-2007/2008 (269 rooms) and semester1-2008/2009 (290 rooms), our proposed approach shows that we are able to produce a solution that satisfies all the constraints (both hard and soft) with a 0 cost value (see Table 1, Column B). Next, using the result from our graph colouring heuristic approach (Kahar and Kendall, 2010), our invigilator scheduling approach is also able to produce a feasible result with no cost value for both of the data sets (see Table 1, Column C).

Based on this result, it is clear that our proposed invigilator scheduling strategy produces a superior solution compared with the solution produced by the UMP proprietary software. We believe that the success of the approach is because of the two-phase method that schedules the chief invigilator followed by the other invigilators. In addition, the ordering of least invigilation duty aids in efficiently selecting suitable invigilators while optimising the spread of invigilation duties (ie, soft constraints, S1 and S2). In discussion with the UMP Academic Office, their poor solution is perhaps due to staff swapping their invigilator duties among themselves after the schedule is published. A common reason being that the invigilator is unsatisfied with their timetable (ie, invigilation duties close to one another, unable to invigilate one (or more) of their own exams is scheduled on the same day, etc) and due to other commitments (eg, meetings, administrative work, etc). The Academic Office will update the changes requested and these changes contribute to a poor solution. Currently, the system neglects the effect of moving or swapping (on request) the invigilation duties, which we will consider in our future work.

We notice that the invigilator scheduling solution depends on the number of rooms being selected in the *exam-timeslot-room* assignment phase. Recall that the total rooms selected from the proprietary software in semester1-2007/2008 and semester1-2008/2009 is 269 and 290, respectively. In our graph colouring (exam timetabling) heuristic (Kahar and Kendall, 2010), the average percentage of rooms selected for semester 1-2007/2008 is 16% (ie, 227) less and for semester 1-2008/2009 it is 10% (ie, 262) less compared with the UMP proprietary software. Obviously, having a lesser number of rooms selected (in the *exam-timeslot-room* assignment phase) would automatically minimise the invigilation duties for the staff.

In summary, we have demonstrated that the proposed invigilator scheduling approach is able to produce a feasible solution that adheres to all constraints without any cost to the objective function (even with a higher number of rooms as in the solution from semester 1 2007/2008 and semester 1 2008/2009). However, the invigilator scheduling result is dependent on the number of rooms being selected from the *exam-timeslot-room* assignment phase.

9. Additional UMP invigilator scheduling constraints

We include additional constraints in addition to the UMP original invigilator constraints as described in Section 5. This is motivated by a survey from Awang *et al* (2006) on the UMP invigilator scheduling problem, which reveal that most of the invigilators are not satisfied with their current invigilation duties. According to Awang *et al*, invigilators suggested that each invigilation duty should have at least a 2- or 3-day gap between them and also suggested having fewer invigilation duties, considering that they also need to be available/on standby during their own exam paper. Additionally, the invigilator requested an even spread of invigilation duties among the staff (as we have considered in the original constraints—see F_1 and F_2 in Section 5). Moreover, according to the timetable officer they often receive request for changes from the invigilators. The common reasons being invigilation duties are consecutive, are to close together, staff need to be on standby as more than one of their exams are scheduled together, etc. We hope to satisfy the invigilators requests and minimise the request for changes to the schedule. The additional hard constraints for the UMP *invigilator scheduling* problem are as follows:

(f) (H6) Invigilators, with a lecturer status, on duty during their exam paper need to be scheduled in the same timeslot and building as their own exam paper. The formulation is as follows:

$$\sum_{p=1}^{R} (v_{lrt} + w_{lrt}) \cdot own\left(a_{il}, x_{it}, y_{ip}\right) = (v_{lrt} + w_{lrt}) \cdot a_{il}x_{it}m_i$$

For all $l \in \{1, \ldots, L\},\ t \in \{1, \ldots, T\}$, $r \in \{1, \ldots, R\}$ and $i \in \{1, \ldots, N\}$ (13)

where

$$\text{own}\left(a_{il}, x_{it}, y_{ip}\right) = \begin{cases} 1 & \text{if } \left(a_{il}x_{it}y_{ip}\right) = 1 \text{ and } \left(B_r = B_p\right) \\ 0 & \text{otherwise} \end{cases}$$

where m_i is the number of rooms, exam i has been split across and B_r is the building for room r. The additional soft constraints are as follows:

(S3) Each invigilation duty should have at least 2-day gap, for every invigilator. A penalty is given if this is violated. The formulation is as follows:

$$F_3 = \sum_{l=1}^{L}\sum_{r=1}^{R}\sum_{t=1}^{T}(v_{lrt} + w_{lrt}) \cdot \left(\sum_{t+s\ (where\ s=1)}^{s \leqslant 5\ and\ t+s \leqslant T}\ \sum_{p=1}^{R} gap\left(v_{lp(t+s)}, w_{lp(t+s)}\right)\right) \tag{14}$$

where

$$\mathrm{gap}\left(v_{lr(t+s)}, w_{lr(t+s)}\right) = \begin{cases} \frac{32}{2^s} & \text{if } \left(v_{lr(t+s)} + w_{lr(t+s)}\right) = 1 \\ 0 & \text{otherwise} \end{cases}$$

(S4) There is a penalty associated with staff on duty during their exam paper. If the staff are on duty during their exam, they need to be scheduled in the same timeslot and building as their exam; see hard constraint, H6.

$$F_4 = \sum_{i}^{N} \sum_{l}^{L} \sum_{r}^{R} \sum_{t}^{T} (v_{lrt} + w_{lrt}) \cdot \mathrm{duty}(a_{il}, x_{it}) \qquad (15)$$

where

$$\mathrm{duty}(a_{il}, x_{it}) = \begin{cases} 3 & \text{if } (a_{il} x_{it}) = 1 \\ 0 & \text{otherwise} \end{cases}$$

10. Results for the additional invigilator constraints

We present the results of the invigilator timetable generated by the UMP proprietary software considering the additional constraint. In our proposed approach, two different strategies were used, which involve sorting the invigilators randomly and also sorting by the least number of invigilation duties (ie, the same as Section 8.3). The results are summarised in Tables 2 and 3.

10.1. Results for the proprietary software

In semester1-2007/2008, considering the additional invigilator constraints, the solution exhibits the following characteristics (see Table 2, Column A).

Hard constraint (H6): The UMP results violate the invigilators on duty during their own exam paper, as they should be assigned in the same timeslot and building as their own exam paper.
Soft constraints: Measuring the solution using the additional soft constraint results in a total of 369 with the cost value for F_3 (Equation (14)) = 120 and F_4 (Equation (15)) = 9 (the value of F_1 and F_2 remain the same).

In semester1-2008/2009, the result shows that (see Table 2, Column A):

- *Hard constraints* (H6): The UMP results violate the constraint.
- *Soft constraints*: The total cost value of the invigilator timetable solution is 713 with F_3 (Equation (14)) = 546 and F_4 (Equation (15)) = 27 (F_1 and F_2 remain the same).

Table 2 Invigilator scheduling results for additional constraint using least duties ordering approach

Constraints	*(A) Proprietary software*		*(B) Our approach using exam timetable from UMP*				*(C) Our approach using exam timetable from* Kahar and Kendall (2010)			
	*Semester*1-2007/2008 (*269 rooms*)	*Semester*1-2008/2009 (*290 rooms*)	*Semester*1-2007/2008 (*269 rooms*)		*Semester*1-2008/2009 (*290 rooms*)		*Semester*1-2007/2008 (*244 rooms*)		*Semester*1-2008/2009 (*274 rooms*)	
			*c*1 ≈39 *s*	*c*5 ≈83 *s*	*c*1 ≈101 *s*	*c*5 ≈180 *s*	*c*1 ≈35 *s*	*c*5 ≈74 *s*	*c*1 ≈90 *s*	*c*5 ≈165 *s*
(H1) Invigilators on duty during their exam paper need to be scheduled in the same timeslot and building as their own exam paper	Not (1)	Not (6)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Total cost value or violation of the soft constraint ($F = F_1$ to F_4)	369	713	978	839	1634	1419	860	86	1092	234

*c*1 = candidates list of one; *c*5 = candidates list of five; Not (*x*) = not comply (number of violations); Yes = comply.

Table 3 Invigilator scheduling results for additional constraint using random ordering approach

Constraints	*(A) Our approach using exam timetable from UMP*				*(B) Our approach using exam timetable from* Kahar and Kendall (2010)			
	*Semester*1-2007/2008 *(269 rooms)*		*Semester*1-2008/2009 *(290 rooms)*		*Semester*1-2007/2008 *(244 rooms)*		*Semester*1-2008/2009 *(274 rooms)*	
	*c*1 ≈39 *s*	*c*5 ≈83 *s*	*c*1 ≈101 *s*	*c*5 ≈180 *s*	*c*1 ≈35 *s*	*c*5 ≈74 *s*	*c*1 ≈90 *s*	*c*5 ≈165 *s*
(H6) Invigilators on duty during their exam paper need to be scheduled in the same timeslot and building as their own exam paper	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Total cost value or violation of the soft constraint ($F=F_1$ to F_4)								
Standard deviation	135	47	143	27	130	31	157	20
Average	2546	310	2886	246	1867	139	227	90
Minimum	2155	201	2578	190	1617	67	1918	49
Maximum	2784	406	3161	306	2180	200	2596	152

*c*1 = candidates list of one; *c*5 = candidates list of five; Not (*x*) = not comply (number of violations); Yes = comply.

10.2. Implementation for the additional invigilator constraints

We consider the additional invigilator constraints in scheduling the invigilators using the *exam-timeslot-room* solution produced by the UMP proprietary software for semester1-2007/2008 and semester1-2008/2009, and the solution from Kahar and Kendall (2010). The following discussion is based on the least invigilation duties ordering and random ordering approach.

10.2.1. Least invigilation duties ordering. The least invigilation duties selects the invigilator with the least duties. In using a candidate list of one for the UMP solutions from semester1-2007/2008 (see Table 2, Column B), our proposed approach shows that we are able to produce a solution that satisfies all the hard constraints with a cost value of 978 (≈39 s). Increasing the candidate list to five, we manage to produce a slightly better solution with a cost value of 839 (≈83 s). For the semester1-2008/2009 data set, using a candidate list of one the solution produced satisfies all the hard constraints with a cost value of 1634 (≈101 s) and with a candidate list of five, the cost value is 1419 (≈180 s). The results are summarised in Table 2, Column B. Comparing the above result with the proprietary software, although our result produces a high cost value (for both data sets), it does satisfy all of the hard constraints compare with the result from the proprietary software.

Next, using the result from our graph colouring heuristic (Kahar and Kendall, 2010) (see Table 2, Column C), for semester1-2007/2008 with a candidate list of one, the solution produced satisfies all the hard constraints with a cost value of 860 (≈35 s). Increasing the candidate list to five, the cost value is 86 (≈74 s), 77% (369 compared with 86 ((369−86)/369 × 100%)) better than the UMP result. For the semester1-2008/2009 data set, using a candidate list of one the solution produced satisfies all the hard constraints with a cost value of 1092 (≈90 s) and with a candidate list of five, the cost value is 234 (≈165 s), that is 67% (713 compared with 234 ((713−234)/713 × 100%)) better than the UMP result. The results are shown in Table 2, Column C. Based on these results, using the approach presented in Kahar and Kendall (2010) to provide the examination timetable, the result we produce is superior to the UMP proprietary solution and also when using the UMP proprietary result, even when we include the additional constraints that are not presented in the proprietary software. We believe the reason for this is that having a lesser number of rooms used (see Table 2), minimises the number of invigilation duties, thus allowing the duties to be spread out more fairly.

10.2.2. Random ordering. In using a candidate list of one on the UMP solutions from semester1-2007/2008, our proposed approach shows that we are able to produce a solution that satisfies all the hard constraints with a minimum cost value of 2155 (see Table 3, Column A). Increasing the candidate list to five, the search produces a far better minimum cost value of 201 that is 45% (369 compared with 201 ((369−201)/369 × 100%)) better when compared with the proprietary software and 76% (839 compared with 201 ((839−201)/839 × 100%)) better when compared with using least duties ordering. For semester1-2008/2009 data set (Table 3, Column A), using a candidate list of one the solution satisfies all the hard constraints with a minimum cost value of 2578. Using a candidate list of five, the minimum cost value is 190, 73% (713 compared with 190 ((713−190)/713 × 100%)) better when compared with the proprietary software and 87% (1419 compared with 190 ((1419−190)/1419 × 100%)) better when compared with using least duties ordering. Referring to the result above, with a candidate list of five, we are able to produce a good quality solution when compared with using a candidate list of one, the UMP proprietary software and using a least duties ordering strategy.

Next, using the result from our graph colouring heuristic (Kahar and Kendall, 2010), for semester1-2007/2008 with a candidate list of one (see Table 3, Column B), the solution produced satisfies all the hard constraints with a minimum cost value of 1617. Increasing the candidate list to five, the solution has a minimum cost value of 67 that is 82% (369 compared with 67 ((369−67)/369 × 100%)) better when compared with the proprietary software and 22% (86 compared with 67 ((86−67)/86 × 100%)) better when compared with using least duties ordering. For semester1-0809 data set (see Table 3, Column B), using a candidate list of one the solution produced satisfies all the hard constraints with a minimum cost value of 1918. Increasing to candidate list of five, the minimum cost value is 49, 92% (713 compared with 49 ((713−49)/713 × 100%)) better when compared with the proprietary software and 79% (234 compared with 49 ((234−49)/234 × 100%)) better when compared with using least duties ordering. Referring to the result above, our proposed approach is able to return a good quality solution (when using a candidate list of five). Overall, the least duties ordering approach produce a good quality solution, outperforming the proprietary software and random ordering (with a candidate list of one). However, the random ordering with a candidate list of five outperforms the least duties ordering approach.

The proposed invigilator scheduling strategy is able to produce good quality solutions even with additional constraints (H6, S3 and S4). This demonstrates that we are able to produce a feasible solution and satisfy the additional invigilator requests (based on the comments of Awang *et al*, 2006), which we believe would benefit the timetable officer (rather than them need to respond to changes post schedule publication). In summary, we have demonstrated that the proposed invigilator scheduling approach is able to produce a feasible solution that adheres to all constraints, including the additional constraints not previously captured.

11. Statement of contribution

This paper has presented a study of a real-world examination timetabling problem from UMP. It involves scheduling invigilators onto an existing exam-timeslot-room solution with the aim of evenly spreading invigilation and chief invigilation duties among the staff. We also include additional constraints, in addition to the current UMP invigilator scheduling constraints. The contributions are as follows:

(a) The collection of invigilator constraints, which have never before been properly documented at UMP.
(b) Formulation of the UMP invigilation scheduling problem as a formal model. The model presented here has never been modelled before in the literature.
(c) Formulation of additional constraints for invigilator scheduling. The additional constraints, we believe more accurately captures the UMP invigilation scheduling problem that is done at the moment.
(d) The use of a simple and efficient constructive technique that produces good quality solutions, satisfying all hard constraints (including the additional constraints) that the UMP proprietary system fails to do.

12. Conclusion and future research directions

In this paper, we have investigated invigilator scheduling for a real-world examination timetabling problem, which aims to satisfy a number of constraints. The problem is complicated by the fact that the chief invigilator position can only be assigned to academic staff and staff are not allowed to invigilate their own papers. Furthermore, the invigilation duties assignment has to meet the required number of invigilators (including the chief invigilator) for each room avoiding clashes and complying with the maximum number of invigilation duties for each member of staff. A least ordering search was used to schedule the invigilators. The proposed approach is able to produce good quality solutions compared with the UMP proprietary software, satisfying all the constraints, both hard and soft, which the proprietary software fails to do. Additionally, we have included extra constraints, based on the comments in Awang *et al* (2006). Different ordering strategies (ie, least duties and random ordering) have been used to schedule the invigilators. We have shown that a good quality solution can be produced even with these additional constraints. We believe that the solutions produced would satisfy all parties (ie, officers and staff).

For future work, our aims are as follows:

- Investigate the optimal number of invigilators required in an examination timetable, which could help to minimise the operational cost instead of selecting non-academic staff (as this takes them away from their other duties).
- Improve on the current invigilator constructive solution using meta-heuristics methodologies.
- Include an automated system that could aid in determining the effect of constraints on the objective value so that the effect of performing swaps can be evaluated. Additionally, it could provide a suggestion (or list of availabilities) in making moves or swapping invigilation duties.

Acknowledgements—The examination/invigilator data set has been provided by the Academic Management Office, UMP and the research has been supported by the Public Services Department of Malaysia (JPA) and the UMP.

References

Awang S, Kahar MNM, Jamil NM and Ajid KA (2006). A satisfaction survey on KUKTEM automated examination scheduling. Malaysian Science and Technology Congress 2006 (MSTC 2006).

Burke EK and Carter MW (eds) (1998). Practice and theory of automated timetabling II, *2nd International Conference, PATAT'97*. Toronto, Canada, 20–22 August 1997, Selected Papers. Lecture Notes in Computer Science, Vol. 1408. Springer, Heidelberg.

Burke EK and De Causmaecker P (eds) (2003). Practice and theory of automated timetabling IV, *4th International Conference, PATAT 2002*. Gent, Belgium, 21–23 August 2002, Selected Revised Papers. Lecture Notes in Computer Science, Vol. 2740 Springer, Heidelberg.

Burke EK and Erben W (eds) (2001). Practice and theory of automated timetabling III, *3rd International Conference, PATAT 2000*. Konstanz, Germany, 16–18 August 2000, Selected Papers. Lecture Notes in Computer Science, Vol. 2079 Springer, Heidelberg.

Burke EK and Ross P (eds) (1996). Practice and theory of automated timetabling, *1st International Conference*. Edinburgh, UK, 29 August–1 September 1995, Selected Papers. Lecture Notes in Computer Science, Vol. 1153. Springer, Heidelberg.

Burke EK and Rudova H (eds) (2007). Practice and theory of automated timetabling VI, *6th International Conference, PATAT 2006*. Brno, Czech Republic, 30 August–1 September 2006, Revised Selected Papers. Lecture Notes in Computer Science, Vol. 3867. Springer, Heidelberg.

Burke EK and Trick M (eds) (2005). Practice and theory of automated timetabling V, *5th International Conference, PATAT 2004*. Pittsburgh, PA, USA, 18–20 August 2004, Revised Selected Papers. Lecture Notes in Computer Science, Vol. 3616. Springer, Heidelberg.

Burke EK, Elliman DG, Ford PH and Weare RF (1996a). Examination timetabling in British Universities—A survey. In: Burke EK and Ross P (eds). The Practice and Theory of Automated Timetabling I: Selected Papers from 1st International Conference on the Practice and Theory of Automated Timetabling. Lecture Notes in Computer Science, Vol. 1153, Springer-Verlag: Edinburgh, UK, pp 76–92.

Burke EK, Newall J and Weare RF (1996b). A memetic algorithm for university exam timetabling. In: *Practice and Theory of Automated Timetabling I*, Lecture Notes in Computer Science, Vol. 1153, Springer, Heidelberg, pp 241–250.

Carter MW and Laporte G (1996). Recent developments in practical examination timetabling. In: Burke EK and Ross P (eds). *Practice and Theory of Automated Timetabling: Selected Papers from the 1st International Conference on the Practice and Theory of Automated Timetabling I*. Lecture Notes in Computer Science, Vol. 1153, Springer-Verlag: Edinburgh, UK, pp 3–21.

Carter MW, Laporte G and Lee SY (1996). Examination timetabling: Algorithmic strategies and applications. *Journal of the Operational Research Society* **47**(3): 373–383.

Cowling P, Kendall G and Hussin NM (2003). A survey and case study of practical examination timetabling problems. In: Burke EK and De Causmaecker P (eds). *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2002)*. Lecture Notes in Computer Science, Vol. 2740, Springer, Heidelberg, pp 258–261.

Dammak A, Elloumi A and Kamoun H (2006). Classroom assignment for exam timetabling. *Advances in Engineering Software* **37**(10): 659–666.

Hussin NM (2005). *Tabu search based hyper-heuristic approaches to examination timetabling*. Phd Thesis, University of Nottingham, UK.

Kahar MNM and Kendall G (2010). The examination timetabling problem at Universiti Malaysia Pahang—Comparison of a constructive heuristic with an existing software solution. *European Journal of Operational Research* **207**(2): 557–565.

Laporte G and Desroches S (1984). Examination timetabling by computer. *Computers & Operations Research* **11**: 351–360.

McCollum B, McMullan PJ, Parkes AJ, Burke EK and Abdullah S (2009). An extended great deluge approach to the examination timetabling problem. In: Blazewicz J, Drozdowski M, Kendall G and McCollum B (eds). *Proceedings of the 4th Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA 2009)*. Dublin, Ireland, pp 424–434.

McCollum B *et al* (2010). Setting the research agenda in automated timetabling: The second international timetabling competition. *INFORMS Journal on Computing* **22**(1): 120–130.

Merlot LTG, Boland N, Hughes BD and Stuckey PJ (2003). A hybrid algorithm for the examination timetabling problem. In: Burke EK and De Causmaecker P (eds). *Practice and Theory of Automated Timetabling IV: Selected Papers from 4th International Conference*. Lecture Notes in Computer Science, Vol. 2740, Springer-Verlag, Heidelberg, pp 207–231.

Ong ML, Liew LH and Sim J (2009). Examination invigilation scheduling system in optimising lecturers' preference. Conference on Scientific & Social Research (CSSR 08'09).

Qu R, Burke EK, McCollum B, Merlot LTG and Lee SY (2009). A survey of search methodologies and automated system development for examination timetabling. *Journal of Scheduling* **12**(1): 55–89.

Reis LP and Oliveira EA (1999). Constraint logic programming approach to examination scheduling. *Artificial Intelligence and Cognitive Science Conference AICS'99*, Cork, Ireland, September.

Received June 2010; accepted December 2011 after one revision