Information Sciences 191 (2012) 146–168

Contents lists available at SciVerse ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

On the use of multi neighbourhood structures within a Tabu-based memetic approach to university timetabling problems

Salwani Abdullah [a,*], Hamza Turabieh [b]

[a] Data Mining and Optimisation Research Group (DMO), Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia

[b] Computer Science Department, Faculty of Science and Information Technology, Zarka University, Jordan

ARTICLE INFO

Article history:
Received 26 July 2011
Received in revised form 21 December 2011
Accepted 28 December 2011
Available online 3 January 2012

Keywords:
Memetic algorithm
Tabu search
University timetabling
Multi neighbourhood

ABSTRACT

Finding a good university timetabling system is not a simple task for a higher educational organisation. As a result, many approaches to generating sufficiently good solutions have been introduced. This is mainly due to the high complexity within the search landscape; moreover, each educational organisation has its own rules and specifications. In this paper, a Tabu-based memetic algorithm that hybridises a genetic algorithm with a Tabu Search algorithm is proposed as an improved algorithm for university timetabling problems. This algorithm is employed on a set of neighbourhood structures during the search process with the aim of gaining significant improvements in solution quality. The sequence of neighbourhood structures has been considered to understand its effect on the search space. Random, best and general sequences of neighbourhood structures have been evaluated in this work. The concept of a Tabu list is embedded to control the selection of neighbourhood structures that are not dependent on the problem domains during the optimisation process after the crossover and mutation operators are applied to the selected solutions from the population pool. The algorithm will penalise neighbourhood structures that are unable to generate better solutions. The proposed algorithm has been applied and evaluated against the latest methodologies in the literature with respect to standard benchmark problems. We demonstrate that the proposed algorithm produces some of the best known results when tested on ITC2007 competition datasets.

1. Introduction

Timetabling problems have long been a challenging area for researchers across the fields of operational research and artificial intelligence. Among the wide variety of timetabling problems, educational timetabling (which covers areas such as school, course and exam timetabling) is one of the most widely studied. This work focuses on course and examination timetabling problems, which can be generally defined as assigning a set of events to a limited number of timeslots and rooms subject to a set of constraints. The problems are adopted from the 2nd International Timetabling Competition (ITC2007) introduced by McCollum et al. [56].

In the past, a wide variety of approaches for solving university timetable problems have been described and discussed in the literature. For a recent detailed overview, readers should consult Qu et al. [66]. Carter [30] categorised the approaches into four types: sequential methods, cluster methods, constraint-based methods and generalised search. Petrovic and Burke [62] added the following categories: hybrid evolutionary algorithms, meta-heuristics, multi-criteria approaches, case-based reasoning techniques, hyper-heuristics and adaptive approaches.

* Corresponding author. Tel.: +603 89216183; fax: +603 89216184.
E-mail addresses: salwani@ftsm.ukm.my (S. Abdullah), turabieh@zu.edu.jo (H. Turabieh).

0020-0255/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2011.12.018

Graph-based techniques are often used to construct initial solutions before improvement techniques are employed. Hybridisation between graph-based techniques and other methods has shown to provide reasonably good results. Some examples of research on graph-based techniques can be found in Carter and Johnson [31], Burke et al. [20], Burke and Newall [27] and Asmuni et al. [13,14]. Other hybridisation approaches are discussed in Abdullah et al. [6,7], and Turabieh and Abdullah [74,76].

Constraint logic programming and constraint satisfaction techniques have also attracted much attention in timetabling due to the ease and flexibility of the employed methods (for example, see [16,36,58]).

Local-search-based techniques, such as Tabu Search and Simulating Annealing, are often used during an improvement phase. Examples of the Tabu Search approach applied to university timetabling problems can be found in Di Gaspero and Schaerf [38], White and Xie [78], and Paquete and Stutzle [61]. Examples of simulated annealing used in university timetabling problems are discussed in Thompson and Dowsland [73], Merlot et al. [58], Burke and Newall [26,27], Burke et al. [18], and Abdullah et al. [9]. Other local-search-based techniques have been described: Kempe chain neighbourhood structures are presented by Casey and Thompson [32], Cote et al. [35], Merlot et al. [58], and Abdullah et al. [8]; multiple neighbourhood structures are presented by Di Gaspero [37]; large neighbourhood search is presented by Abdullah et al. [1,2]; variable neighbourhood search is presented by Mladenovic and Hansen [59], Hansen and Mladenovic [47], Burke et al. [22,25], Ahmadi et al. [12]; greedy randomised adaptive search approach is presented by Casey and Thompson [32]; iterated local search is presented by Lourenco et al. [51]; great deluge algorithm is presented by McCollum et al. [57]; and a fine-tuned local search method is presented by Caramia et al. [29].

Population-based algorithms, such as genetic algorithms and evolutionary algorithms, have also been studied in relation to university timetabling research; for example, see Corne et al. [34], Ross et al. [67,68], Terashima-Marin et al. [72], Erben [42], Cote et al. [35] and Ulker et al. [77]. An extension of genetic algorithms often referred to as a memetic algorithms allow for individuals within a population to be improved within a generation. The implementation of memetic algorithms to solve university timetabling problems is discussed in Burke et al. [19,20], Eley [41], Ersoy et al. [43], Abdullah et al. [4,7], Abdullah and Turabieh [5], and Turabieh and Abdullah [74]. Other methods that are categorised as population-based algorithms are ant algorithms (see Dowsland and Thompson [40]), artificial immune algorithms (see [61,54]), honey bees algorithms (see [70]), and fish swarm optimisation algorithms (see [74,75]).

Hyper-heuristic-based approaches are becoming increasingly popular in the recent literature (Qu and Burke. [65]). Hyper-heuristics can be considered *heuristics that choose heuristics* and have attracted increasing levels of research attention with the aim of developing more general approaches (Burke et al. [24]). Examples of studies that have employed hyper-heuristics to university timetabling problems include Ahmadi et al. [10], Ross et al. [69], Kendall and Hussin [48], Burke et al. [21–23], Bilgin et al. [15], Ersoy et al. [43], and Qu and Burke [64].

Decomposition and clustering techniques have also been tested on timetabling problems. For example, Lim [50] developed a multi-agent algorithm in which problems are divided into sub-problems and locally solved by an agent; additionally, Qu and Burke [64] investigated an approach in which events are adaptively decomposed into *difficult* and *easy* sets.

The above-mentioned approaches have been tested on various universities timetabling datasets that can be found at http://www.asap.cs.nott.ac.uk/resource/data. Due to the existence of multiple formulations of university timetabling problems, the 2nd International Timetabling Competition (ITC2007) [56] attempted to standardise the problems found within educational timetabling by introducing three tracks (one on exam and two on course timetabling) in which the problems incorporated more real-world constraints. In doing so, the organisers attempted to reduce the acknowledged gap between research and practice that exists in this area [55].

A brief observation of the recent timetabling literature shows that most of meta-heuristic methods use small neighbourhood structures in their search algorithms. Research also has shown that hybridised meta-heuristic methods often perform better than individual approaches, as they are benefited from the advantages of both (or more) techniques. This leads to the investigation of a hybrid method that uses large neighbourhood structures to solve the timetabling problem. The approach maintains the importance of neighbourhood structures and understands its effect on the search space.

Interested readers can find more details about university timetabling research in the comprehensive survey paper by Qu et al. [66] and Lewis [49].

The rest of the paper is organised as follows. The next section formally presents the university timetabling problem. The solution approach is outlined in Section 3. Our results are presented and discussed in Section 4. An analysis of our results is presented in Section 5, which is followed by brief concluding comments in Section 6.

2. Problem description

In this work, we solved two different tracks of the 2nd International Timetabling Competition (ITC2007), i.e., examination timetabling and curriculum-based course timetabling problems. These two tracks (Track 1 and Track 3) are selected because Track 1 represents the examination timetabling problem, which is quite difficult or complex compared to other problems, such as the uncapacitated examination timetabling problem (see [31]). Track 3 represents curriculum-based course timetabling problems, which are quite similar to real-world scenarios. These two tracks are chosen because they represent two distinct domains with different soft and hard constraints. Track 2 is not considered because it represents almost the same problem as Track 3 in terms of the hard and soft constraints. Furthermore, the hard constraints in Track 2 (with 3 hard

constraints) are a subset of the hard constraints of Track 3 (with 4 hard constraints), which in turn can be recognised as a less complex problem. The following subsections discuss both tracks considered in this work.

2.1. Problem A: curriculum-based course timetabling

The curriculum-based course timetabling problem constitutes the third track of the 2nd International Timetabling Competition (ITC2007). The main reason why this formulation is so widely accepted is because it simulates a set of real problems that have been collected from several universities throughout the world. This problem focuses on a weekly assignment of a set of lectures to specific timeslots and rooms, where conflicts between courses are set according to curricula published by the university and are not based on enrolment data.

In this paper, we consider the same curricula-based course timetabling problem as described in Di Gaspero et al. [39], which consists of the following entities: Days, Timeslots and Periods. We are given the number of teaching days between weeks 5 and 6. The main characteristics for this problem are listed below.

- "Each day is divided into a fixed number of timeslots, which is equal for all days."
- "A period is a pair composed of a day and a timeslot. The product of the days and the timeslots represent the total number of periods."
- "Each course consists of a fixed number of lectures to be scheduled in distinct periods which is taught by a teacher and attended by a number of students. Each course has a minimum number of days within which the lecture for that particular course should be spread. There are some periods in which the course cannot be scheduled."
- "Each room has a capacity and location. Capacity is represented in terms of available seats. Location is represented as an integer value corresponding to a separate building. Some rooms are not suitable for some courses due a lack of required equipment."
- "A curriculum represents a group of courses such that any pair of courses in the group has students in common. Based on curricula, we have the conflicts between courses and other soft constraints."

The solution to the problem is an assignment of a period (day and timeslot) and a room to all lectures of each course. The hard constraints shown in Table 1 have been borrowed from Di Gaspero et al. [39]:

The soft constraints shown in Table 2 have also been borrowed from Di Gaspero et al. [39].

The curriculum-based timetabling problem is composed of a set of n courses $C = \{c_1, \ldots, c_n\}$ to be assigned to a set of m rooms $R = \{r_1, \ldots, r_m\}$ and a set of p periods $T = \{t_1, \ldots, t_p\}$. Every course $c_i$consists of l_i lectures to be scheduled. A period consists of a day and a timeslot. A total of p periods are spread out over h daily timeslots and d days, i.e., $p = h \times d$. In addition, there are a set of w curricula $CR = \{Cr_1, \ldots, Cr_w\}$ in each curriculum Cr_k is a group of courses that share common students. A candidate solution is represented by a $p \times m$ matrix S, where corresponds to the course label assigned to room r_j and period t_i. Table 3.4 shows a list of symbols and variable definitions for curricula, courses, rooms, and solution X that are implemented in this work. Note that the formulation stated in Table 3 is adopted from Lu and Hao [52].

Hard constraints and the penalty cost for the soft constraints are given as follows:

- H1: *Lectures*: $\forall c_k \in C$,

$$\sum_{i=1,\ldots,p, j=1,\ldots,m} \chi\{x_{i,j} = c_k\} = l_k$$

Where χ is a truth indicator function which takes value of 1 if the given proposition is true, and 0 otherwise.

- H2: *Conflicts*: This hard constraint is always satisfied using our solution representation.

$$\forall x_{i,j}, x_{i,k} \in X, \quad x_{i,j} = c_u, \quad x_{i,k} = c_v,$$
$$con_{uv} = 0$$

Table 1
Hard constraints for curriculum-based course timetabling.

Hard constraints	Explanation
H1 (Lectures)	"All lectures of a course must be scheduled, and assigned to distinct periods. A violation occurs if a lecture is not scheduled or two lectures within a course are scheduled in the same period."
H2 (Conflicts)	"All lectures of courses in the same curriculum or taught by the same teacher must be scheduled in different periods. Two conflicting lectures in the same period represent one violation. Three conflicting lectures count as 3 violations: one for each pair."
H3 (Room occupation)	"Two lectures cannot be assigned to the same room at the same period. Two lectures in the same room at the same period represent one violation. Any extra lecture in the same period and room counts as one more violation."
H4 (Availability)	"If the teacher of the course is not available to teach that course at a given period, then no lecture of the course can be scheduled at that period. Each lecture scheduled in a period unavailable to that course is one violation."

Table 2
Soft constraints for curriculum-based course timetabling.

Soft constraints	Explanation
S1 (Room capacity)	"The number of students that attend the course for each lecture must be less than or equal to the number of seats of the rooms hosting its lectures. Each student above the capacity counts as 1 violation."
S2 (Minimum working days)	"The lectures of each course must be spread over the given minimum number of days. Each day below the minimum, counts as 1 violation."
S3 (Isolated lectures)	"Lectures belonging to a curriculum should be adjacent to each other (i.e., in consecutive periods). For a given curriculum we account for a violation every time there is one lecture not adjacent to any other lecture within the same day. Each isolated lecture in a curriculum counts as 1 violation."
S4 (Room stability)	"All lectures of a course should be delivered in the same room. Each distinct room used for the lectures counts as 1 violation."

- H3: *Room occupancy*: $\forall x_{k,j} \in X, x_{k,j} = c_i$;
- H4: *Availability*: $\forall x_{i,j} = c_k \in X\ uav_{k,i} = 0$.
- S1: *Room capacity*: $\forall x_{i,j} = c_k \in X$,

$$f_1(x_{i,j}) = \begin{cases} std_k - cap_j, & \text{if } std_k > cap_j, \\ 0, & \text{otherwise}. \end{cases}$$

- S2: *Minimum working days*: $\forall c_i \in C$,

$$f_2(C_i) = \begin{cases} md_i - nd_i(X), & \text{if } nd_i(X) > md_i, \\ 0, & \text{otherwise}. \end{cases}$$

- S3: *Isolated lectures* (*curriculum compactness*): $\forall x_{i,j} = c_k \in X$,

$$f_3(x_{i,j}) = \sum_{Cr_q \in CR} \chi\{c_{+k} \in Cr_q\} \cdot iso_{q,i}(X),$$

where

$$iso_{q,i}(X) = \begin{cases} 1, & \text{if } (i \bmod h = 1 \vee app_{q,i-1}(X) = 0) \wedge (i \bmod h = 0 \vee app_{q,i+1}(X) = 0), \\ 0, & \text{otherwise}. \end{cases}$$

- S4: *Room stability*: $\forall c_i \in C$,

$$f_4(c_i) = nr_i(x) - 1.$$

Based on the formulation as stated above, the penalty cost for current solution X is identified in formula below. Then, the goal is to find a feasible solution X^* such that $f(X^*) < f(X)$ for all X in the feasible search space.

$$f(X) = \sum_{x_{i,j} \in X} \alpha_1 \cdot f_1(x_{i,j}) + \sum_{c_i \in C} \alpha_2 \cdot f_4(c_i) + \sum_{c_i \in C} \alpha_3 \cdot f_2(c_i) + \sum_{x_{i,j} \in X} \alpha_4 \cdot f_3(x_{i,j}),$$

α_1, α_2, α_3 and α_4 are the penalties associated with each of the soft constraints. In the CB-CTT formulation, they are set as follows: $\alpha_{1=1}$, $\alpha_{2=1}$, $\alpha_{3=5}$ and $\alpha_{4=2}$. Note that $\alpha_1 \sim \alpha_4$ are fixed in the problem formulation and should not be confused with the penalty parameters used by some solution procedures (Lu and Hoa [52]).

Twenty-one real-world instances have been tested in this work. These instances are provided by the University of Udine. The main features of the instances used are given in Table 4. The details of all instances can be found at http://tabu.diegm.uniud.it/ctt/index.php.

2.2. Problem B: examination timetabling problem

The examination timetabling problem considered in this problem is taken from the first track of the 2nd International Timetabling Competition (ITC2007) (http://www.cs.qub.ac.uk/itc2007/index.htm); eight cases have been introduced, which represent a particular formulation of examination timetabling. A set of hard and soft constraints are adopted to reflect real-world problems. The hard and soft constraints considered in this problem are listed below. The hard constraints are shown in Table 5.

The soft constraints are shown in Table 6.

The objective function is to minimise the violation of the soft constraints as shown in the following equation [39]:

$$\min \sum_{s \in S} (w^{2R} C_S^{2R} + w^{2D} C_S^{2D} + w^{PS} C_S^{PS}) + w^{NMD} C_s^{2NMD} + w^{FL} C^{FL} + w^{p} C^{P} + w^{R} C^{R} \quad (1)$$

Each dataset has its own weight, as shown in Table 7 (McCollum et al. [56]).

A feasible timetable is one in which all examinations have been assigned to a period and room and no violation of the hard constraints.

Table 3
Notations used for the curriculum-based course timetabling problem.?

Symbols	Description
n	total number of courses
m	total number of rooms
d	number of working days per week
h	number of timeslots per working day
p	total number of periods, $p = d \times h$
s	total number of curricula
C	set of courses, $C = \{c_1,\ldots,c_n\}$, $\|C\| = n$
R	set of rooms, $R = \{r_1,\ldots,r_m\} = m$
T	set of periods, $T = \{t_1,\ldots,t_p\}$, $\|T\| = p$
CR	set of curricula, $CR = \{Cr_1,\ldots,Cr_w\}$, $\|CR\| = w$
Cr_k	kth curriculum including a set of courses
l_i	number of lectures of course c_i
l	total number of all lectures, $l = \sum_{1}^{n} l_i$
std_i	number of students attending course c_i
tc_i	teacher instructing course c_i
md_i	number of minimum working days of course c_i
cap_j	capacity of room r_j
$uav_{i,j}$	whether course c_i is unavailable at period t_j. $uav_{i,j} = 1$ if it is unavailable, $uav_{i,j} = 0$ otherwise
con_{ij}	whether course c_i and c_j are conflict with each other; $con_{ij} = \begin{cases} 0, & \text{if } (tc_i \neq tc_j) \wedge (\forall Cr_q, c_i \notin) Cr_q \wedge c_j \notin Cr_q \\ 1, & \text{otherwise} \end{cases}$
$x_{i,j}$	the course scheduled at room r_j and period t_i
$nr_i(X)$	number of rooms occupied by course c_i for a candidate solution X; $nr_i(X) = \sum_{j=1}^{m} \sigma_{ij}(X)$ where $\sigma_{ij}(X) = \begin{cases} 1, & \text{if } \forall x_{k,j} \in X, \quad x_{k,j} = c_i, \\ 0, & \text{otherwise} \end{cases}$
$nd_i(X)$	number of working days that course c_i takes place at in candidate solution X, $nd_i(X) = \sum_{j=1}^{d} \beta_{ij}(X)$ where $\beta_{ij}(X) = \begin{cases} 1, & \text{if } \forall x_{u,v} \in X, \quad x_{u,v} = c_i \wedge \left[\frac{u}{h}\right] = j, \\ 0, & \text{otherwise} \end{cases}$
$app_{k,i}(X)$	whether Cr_k appears at t_i in X; $app_{k,i}(X) = \begin{cases} 1, & \text{if } \forall x_{i,j} \in X, \quad x_{i,j} = c_u \wedge c_u \in Cr_k, \\ 0, & \text{otherwise} \end{cases}$

3. The Tabu-based memetic approach

Memetic algorithms have been widely applied in different fields of optimisation because of their ability to explore the search space better than standard evolutionary algorithms or local search algorithms [19,20].

3.1. Chromosome representation

A simple chromosome representation can improve the crossover (timeslot crossover) and mutation operations based on the feasibility of the structure after performing genetic algorithm operations [63,33]. In this work, we use a simple type of chromosome representation for university timetabling problems. Fig. 1 shows an example of a chromosome representation (for Problem B) composed of a string of genes. Each chromosome represents a feasible solution, where the gene represents timeslot t_i, room r_i, and event e_i. For example, events e_1, e_5, e_7, e_{10} are scheduled during timeslot t_1 in rooms r_1, r_2, r_3 and r_4, respectively.

The chromosome representation for Problem A is shown in Fig. 2. The solution of the problem is an assignment of a period (day (d) and timeslot (t)) and a room (r), to all lectures of each course (c). For example, course c_5 is scheduled during period 1 (day d_1, timeslot t_2) in room r_1. Course c_8 is scheduled during period 2 (day d_1 and timeslot t_2) and room r_1, etc.

Table 4
Features of curriculum-based course timetabling instances.

Instance	Course	Total lectures	Rooms	Period per day	Days	Curricula	Min and max lectures per day per curriculum
comp01	30	160	6	6	5	14	2–5
comp02	82	283	16	5	5	70	2–4
comp03	72	251	16	5	5	68	2–4
comp04	79	286	18	5	5	57	2–4
comp05	54	152	9	6	6	139	2–4
comp06	108	361	18	5	5	70	2–4
comp07	131	434	20	5	5	77	2–4
comp08	86	324	18	5	5	61	2–4
comp09	76	279	18	5	5	75	2–4
comp10	115	370	18	5	5	67	2–4
comp11	30	162	5	9	5	13	2–6
comp12	88	218	11	6	6	150	2–4
comp13	82	308	19	5	5	66	2–3
comp14	85	275	17	5	5	60	2–4
comp15	72	251	16	5	5	68	2–4
comp16	108	366	20	5	5	71	2–4
comp17	99	339	17	5	5	70	2–4
comp18	47	138	9	6	6	52	2–3
comp19	74	277	16	5	5	66	2–4
comp20	121	390	19	5	5	78	2–4
comp21	94	327	18	5	5	78	2–4

Table 5
Hard constraints for examination timetabling problems.

Hard constraints	Explanation
H1	No student sits more than one examination at a time
H2	The capacity of individual rooms is not exceeded at any time throughout the examination session
H3	Period duration restrictions are not violated
H4	Period related hard constraints, e.g., Exam_A must be placed after Exam_B
H5	Room related hard constraints, e.g., Exam_A must use Room 101

Table 6
Soft constraints for examination timetabling problems.

Soft constraints	Mathematical symbol	Explanation
S1	C_S^{2R}	Student has to sit two exams in a row (adjacent on same day)
S2	C_S^{2D}	Student has to sit two exams in a day
S3	C_S^{PS}	Student does not have a specified spread (in terms of periods) of examinations
S4	C_S^{2NMD}	Mixed durations of examinations occur within individual periods
S5	C^{FL}	Examinations of large class sizes appear later in the examination session
S6	C^{P}	Period-related soft constraints
S7	C^{R}	Room-related soft constraints

Table 7
The associate weight of ITC2007 collection of examination datasets.

Data sets	w^{2D}	w^{2R}	w^{PS}	w^{NMD}	w^{FL}	w^{P}	w^{R}
Exam1	5	7	5	10	100	30	5
Exam2	5	15	1	25	250	30	5
Exam3	10	15	4	20	200	20	10
Exam4	5	9	2	10	50	10	5
Exam5	15	40	5	0	250	30	10
Exam6	5	20	20	25	25	30	15
Exam7	5	25	10	15	250	30	10
Exam8	0	150	15	25	250	30	5

3.2. The memetic operators: crossover and mutation

The crossover operation is illustrated in Fig. 3, which represents a period-exchange crossover. We adopt the same crossover mechanism described in Cheong et al. [33]; however, instead of the standard crossover operators the period-exchange crossover operator is able to preserve a favourable temporal relationship between events. This crossover mechanism is carried out under two conditions to ensure that the feasibility of offspring is maintained.

1. An event can be moved only if the corresponding timeslot is empty. For example, e_6 can be moved from timeslot t_2 in parent (a) to timeslot t_8 in child (b).
2. No conflicts occur between moved events and scheduled events. For example, in child (a), there should be no conflict between moved event e_8 from t_8 (parent (b)) and scheduled events e_6, e_7 in t_2 (parent (a)).

The shaded periods represent the selected for a crossover operation. These periods are selected based on a crossover rate using a roulette wheel selection method. For example, timeslots t_2 and t_8 are chosen as parent (a) and parent (b), respectively. Crossover is performed by inserting all events from timeslot t_8 in parent (b) to timeslot t_2 in parent (a), which then produced a child (a). The same process is applied to obtain child (b). This operation leads to an infeasible solution due to a conflict appeared between a number of events. For example, in child (a), $e_8$and e_{22} are repeated twice in child (a) in t_3 and t_8, respectively. The duplication should be removed to ensure that child (a) is feasible (Note that e_8 and e_{22} should not conflict with e_6 or e_7). In child (b), e_6 is duplicated in t_8 and t_9. As a result, it should be removed from t_9 to ensure the feasibility. Note that the event should only be scheduled once throughout the exam period. The most likely hard constraint that will be violated is H1 (see Table 5).

Fig. 4 shows an example of the period-exchange crossover for Problem A (ITC2007-Track 3).

Again, the shaded periods represent the periods selected for a crossover operation, i.e., timeslots t_4 and t_2 are chosen as parent (a) and parent (b), respectively, to produce child (a) and child (b). This operation leads to an infeasible solution of

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
r_1	e_1	e_{21}	.	.	.	.	.	.	e_{24}
r_2	e_5	e_{18}	.	.	.	.	.	.	e_9
r_3	e_7	e_{26}	.	.	.	.	.	.	e_{13}
r_4	e_{10}	.	.	.	.	.	.	.	.

Fig. 1. Chromosome representation for ITC2007-Track 1 (Problem B).

	d1				*d2*				*d3*				*d4*				*d5*			
	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4
r_1		c_8	.	c_{10}	c_1	c_{11}	.	.	c_1	c_3	.	c_{13}	c_1	c_{10}	c_5	c_{11}	c_1	c_3	.	.
r_2	c_5	c_{11}	c_7	c_3	c_7	.	c_5	c_8	c_5	c_{11}	c_9	c_7	c_8	c_4	c_{13}		c_5	c_{11}	c_8	c_9
r_3	c_{13}	.	c_1	.	c_{10}	.	c_{13}	.	.	.	c_2	.	c_7	.	.	.	c_7	c_{10}	c_{13}	.

Fig. 2. Chromosome representation for ITC2007-Track 3 (Problem A).

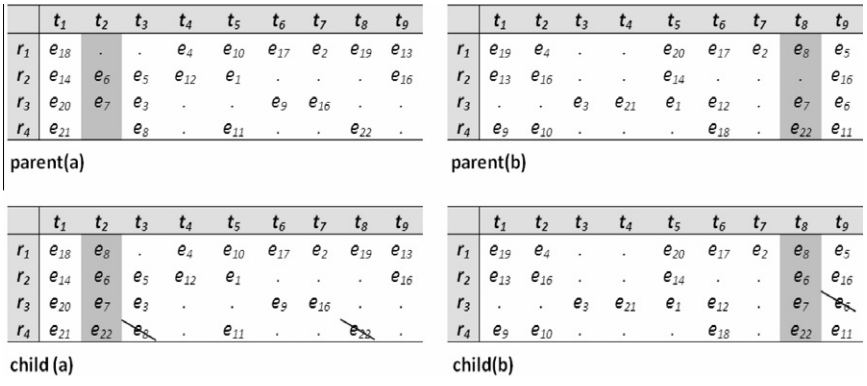

Fig. 3. Chromosome representations after crossover for ITC2007-Track 1 (Problem B).

child (a), where c_{11} is scheduled twice during the same timeslot t_2 but in different rooms, i.e.,r_2 and r_3. Note that, the same teacher cannot handle two classes at the same time. The most likely hard constraints that will be violated after the crossover operation (for this example) are H2 and H4 (see Table 1). Thus, c_{11} (which is scheduled in r_2 or r_3) should be removed to ascertain the feasibility of the timetable. Moreover, the number of classes for each event should not increase throughout the crossover operation. For example, event X has three classes in a week time. After a crossover, maybe the number of classes for event X now becomes4 because of the duplication. So, in order to maintain the same number of classes for each event, the duplication must be removed. Based on the example in Fig. 4, c_{12} is moved from parent (b) to parent (a). As a result, the number of occurrence for c_{12} is now increased. In order to ensure the feasibility, select c_{12} at random and remove from child (a). The same operation is carried out for events c_8 and c_{11} of child (b).

The mutation is used to enhance the performance of the crossover operation by allowing a large search space to be explored and to maintain the genetic diversity from one generation of a population to the next [71]. The new solution is then mutated based on the mutation rate (see Table 8 for parameter settings) by moving/swapping some events from one timeslot to another. In this work, the mutation operator functions as follows: First, select randomly one neighbourhood structure and apply it to the current solution. Then, accept the new solution regardless of its quality. We use "neighbourhood structure" because the modification made in the mutation phase is similar to the idea of a neighbourhood structure, which is moving or swapping some events from one timeslot to another to modify the solution. Generally, the term "neighbourhood structure" is widely used in the local search algorithm to generate a neighbour solution by perturbing the current one. In a local search algorithm however, only a solution that is better than the current one or satisfies the acceptance criterion will be accepted. Fig. 5a and b represent the examination timetabling solution before and after the mutation operator takes place.

Based on the example in Fig. 5a and a mutation rate (note that the mutation rate used in this work is presented in Table 8), a number of events are chosen at random and are moved/swapped to a random timeslot/room while maintaining the feasibility of the solution. For example, e_{24} (that is scheduled at t_9 in r_1) is selected at random and is moved to a random timeslot (assume t_5) and room (assume r_3), and e_{26} (at t_2 in r_3) is swapped with e_9 (at t_9 in r_2) as shown in Fig. 5b.

3.3. The algorithm

In this paper, a Tabu-based memetic algorithm is proposed to solve university timetabling problems, where a large neighbourhood structures have been used as a local search mechanism. The aim of using large neighbourhood structures inside genetic operators is to provide significant improvements in the quality of the solution.

parent(a)

	d1				*d2*				*d3*				*d4*				*d5*			
	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4
r_1		c_8	.	c_{10}	c_1	c_{11}	.	.	c_1	c_3	.	c_{13}	c_1	c_{10}	c_5	c_{11}	c_1	c_3	.	.
r_2	c_5	c_{11}	c_7	c_3	c_7	.	c_5	c_8	c_5	c_{11}	c_9	c_7	c_8	c_4	c_{13}		c_5	c_{11}	c_8	c_9
r_3	c_{13}	.	c_1	.	c_{10}	.	c_{13}	.	.	.	c_2	.	c_7	.	.	.	c_7	c_{10}	c_{13}	.

parent(b)

	d1				*d2*				*d3*				*d4*				*d5*			
	t_1	t_3	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4
r_1	c_{17}	c_7	c_{10}	c_{14}	c_3	.	c_{19}	c_7	c_{11}	.	c_7	c_1	c_1	c_1	c_4		c_3	c_1	c_5	c_2
r_2	c_8	c_5	c_1	c_{15}	c_1	c_8	c_5	.	c_5	c_{13}	.		c_{13}	c_{10}	.		.	.	c_8	c_6
r_3	c_{11}	.	c_{12}	.	c_{10}	c_{16}	c_{14}	c_5	c_9	c_8	c_2	.	c_4	c_7	c_{13}	c_{12}	c_7	c_{17}	c_{11}	.

child(a)

	d1				*d2*				*d3*				*d4*				*d5*			
	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4
r_1			.	c_{10}	c_1	c_{11}	.	.	c_1	c_3	.	c_{13}	c_1	c_{10}	c_5	c_{11}	c_1	c_3	.	.
r_2	c_5		c_7	c_3	c_7	.	c_5	c_8	c_5	c_{11}	c_9	c_7	c_8	c_4	c_{13}		c_5	c_{11}	c_8	c_9
r_3	c_{13}	c_{12}	c_1	.	c_{10}	.	c_{13}	.		.	c_2	.	c_7	.	~~c_{12}~~	.	c_7	c_{10}	c_{13}	.

child(b)

	d1				*d2*				*d3*				*d4*				*d5*			
	t_1	t_3	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4
r_1	c_{17}	c_7	c_{10}	c_{14}	c_3	.	c_{19}	c_7	c_{11}	.	c_7	c_1	c_1	c_1	c_4	c_8	c_3	c_1	c_5	c_2
r_2	c_8	c_5	c_1	c_{15}	c_1	~~c_8~~	c_5	.	c_5	c_{13}	.	.	c_{13}	c_{10}	.	c_{11}	.	.	c_8	c_6
r_3	~~c_{11}~~	.	c_{12}	.	c_{10}	c_{16}	c_{14}	c_5	c_9	c_8	c_2	.	c_4	c_7	c_{13}	c_{12}	c_7	c_{17}	c_{11}	.

Fig. 4. Chromosome representations after crossover for ITC2007-Track 3 (Problem A).

Table 8
Parameters setting.

Parameter	Value
Population size	50
Crossover rate	0.8
Mutation rate	0.04
Tabu list size	3

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
r_1	e_1	e_{21}	.	.	.	.	.	.	e_{24}
r_2	e_5	e_{18}	.	.	.	.	.	.	e_9
r_3	e_7	e_{26}	.	.	.	.	.	.	e_{13}
r_4	e_{10}	.	.	.	.	.	.	.	.

(a) Before mutation

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
r_1	e_1	e_{21}	.	.	.	.	.	.	.
r_2	e_5	e_{18}	.	.	.	.	.	.	e_{26}
r_3	e_7	e_9	.	.	e_{24}	.	.	.	e_{13}
r_4	e_{10}	.	.	.	.	.	.	.	.

(b) After mutation

Fig. 5. Solution representations before and after mutation for ITC2007-Track 3 (Problem A).

Note that we have conducted the comparison between the Tabu-based memetic algorithm and memetic algorithm alone on the same instances. Experimental results show that the Tabu-based memetic algorithm outperforms the memetic algorithm alone. This motivates us to investigate further the performance of the memetic algorithm with the Tabu mechanism.

3.3.1. Neighbourhood structures

In our previous work reported in Abdullah et al. [3], in which a list of neighbourhood structures was applied within the iterative improvement algorithm tested on post enrolment course timetabling problems, we found that the proposed approach was able to obtain some of the best results (at the time the paper was published). Based on this experiment, we have adapted the same neighbourhood structures and employed them within the Tabu-based memetic algorithm. The different neighbourhood structures and their explanations can be outlined as follows (adapted from Abdullah et al. [3]):

Nbs1: Select two events at random and swap timeslots.
Nbs2: Choose a single event at random and move to a new random feasible timeslot.
Nbs3: Select two timeslots at random and simply swap all the events in one timeslots with all the events in the other timeslots.
Nbs4: Take two timeslots at random, say t_i and t_j (where $j > i$), where timeslots are ordered $t_1, t_2, t_3, \ldots, t_p$. Take all events in t_i and allocate them to t_j; then allocate those that were in t_{j-1} to t_{j-2} and so on until we allocate those that were t_{j+1} to t_i and terminate the process.
Nbs5: Move the highest penalty event from a random 10% selection of the events to a random feasible timeslot.
Nbs6: Carry out the same process as in *Nbs5* but with 20% of the events.
Nbs7: Move the highest penalty event from a random 10% selection of the events to a new feasible timeslot that can generate the lowest penalty cost.
Nbs8: Carry out the same process as in *Nbs7* but with 20% of the events.
Nbs9: Select two timeslots based on the maximum enrolled events, say t_i and t_j. Select the most conflicted event in t_i and t_j, and then apply a kempe chain from Thompson and Dowsland [73].

A schematic overview of the algorithm is presented in Fig. 6, where the populations are generated (at first) using the saturation degree heuristic (see [17]). For more details about applying graph colouring to timetabling (see [28]). Two parents (solutions) are selected from the population-based on the roulette wheel selection (RWS) procedure. Two memetic operators such as crossover and mutation are employed prior to the improvement algorithm (i.e., Tabu-based memetic algorithm). A set of neighbourhood structures have been employed within an improvement algorithm. The best solution found after the

employment of the improvement algorithm will be added later to the population pool while maintaining the size of the population.

Fig. 7 shows the pseudo code for the proposed approach. The algorithm begins by creating an initial population. The best solution from the population, S_{best}, is selected. A Tabu list with the size $Tabu_{Size}$ is created to prevent ineffective neighbourhood structures from being selected in the next iteration and provide a better opportunity for the remaining neighbourhood structures to be explored (see Table 8 for parameter setting). A variable *CanSelect* (represents a boolean value) allows the algorithm to control the selection of a neighbourhood structure. For example, if a neighbourhood structure, *Nbs*, exhibits good performance (in terms of producing a lower penalty solution), the algorithm will continue to apply *Nbs* in the next iteration until no good solution can be obtained. In this case, *Nbs* will be pushed into the FIFO (First In First Out) Tabu list, T_{List}. This move is not allowed to be part of any search process for a certain number of iterations, which is equal to the Tabu size. In a *do-while* loop, two solutions are randomly selected, S_1 and S_2. The crossover and mutation operators are applied to S_1 and S_2 to obtain S_1' and S_2'. A neighbourhood structure, *Nbs*, is randomly selected and is applied to S_1' and S_2' to obtain S_1'' and S_2''. The best solution among S_1', S_2', S_1'' and S_2'' is chosen and assigned to a current solution S^*. If S^* is better than the best solution at hand, S_{best}, then S^* is accepted. Otherwise, *Nbs* will be added to T_{List}. The members of the populations

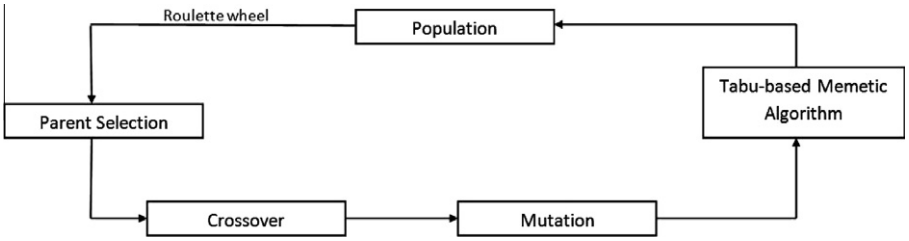

Fig. 6. Tabu-based memetic algorithm.

do while (i<population size)
  Generate random solutions, S_i;
end while
Set the length of the tabu, *TabuSize*;
Choose a best solution from the population, S_{best};
Create an empty tabu list with *TabuSize*, T_{List};
Set *CanSelect* ← *True*
do while (not termination criterion)
  Select two parents from the population using a roulette wheel selection, S_1 and S_2;
  Apply crossover and mutation operators on S_1 and S_2 to produce S_1' and S_2';
  if *CanSelect*== *True*
    Select a neighbourhood structure, which is not in T_{List}, called *Nbs*;
  end if
  Apply *Nbs* on S_1' and S_2' to produce S_1" and S_2";
  Obtain a minimum solution penalty from S_1', S_2', S_1" and S_2", called current solution S*;
  if (S* <S_{best})
    S_{best}← S*;
    CanSelect ← *False*;
  else
    Push *Nbs* to T_{List};
    CanSelect ← *True*;
  end if
  Update the sorted population by inserting and removing the best and worst solutions,respectively, while maintaining the size of the population;
end while
Output the best solution, S_{best};

Fig. 7. Tabu-based memetic algorithm.

will be updated by removing the worst solution and inserting the new solution obtained from the search process while maintaining the size of the population to be used in the next generation. The process is repeated until the termination criterion is met (in this work the termination criterion is set as the computational time).

4. Experimental results

The proposed algorithms are programmed using Matlab, and simulations are performed on an Intel Pentium 4 2.33-GHz computer. We ran the experiments for 600 s based on ITC2007 competition rules [39,56]. The parameter settings are listed in Table 8.

The parameter settings were carefully selected based on our preliminary experiments over university timetabling problems. These settings should not be regarded as the optimal set of parameter values but should be regarded as a generalised set of parameter values because the proposed algorithm performs fairly well over the test problems.

In this work, the size of the population was set to 50, the mutation rate to 0.04, and the crossover rate to 0.8. We used an acceptable number of populations due to the termination criterion that has been set in these datasets (i.e., computational time of 600 s) and to provide the solution in the population a better probability of perturbing. This, in turn, provided a better opportunity to search the solution space. The role of crossover operators is to inherit some characteristics of the two parents to generate offspring. The most commonly used rates are in the interval [0.45, 0.95] (see [71]). At a 0.8 crossover rate, we believe the proposed algorithm is able to obtain strong heritability from both selected parents. A low mutation rate was used in this experiment to have a minimal impact (diversification) on the individuals selected from the population while maintaining valid solutions, i.e., this will affect the quality of the solution when a move (perturbation) is performed. For small changes made on the meme, the solution should also reveal small changes. Otherwise, if small changes produce a large effect on the solution, then the search will converge toward a random search in the landscape [71].

We perform three types of experiments as follows:

1. "**Random**" neighbourhood selection: This is a random selection of neighbourhood structures. A new neighbourhood structure is selected at random if the previous one no longer improves the quality of the current solution.
2. "**Best**" sequence of neighbourhood selection: This sequence is obtained from the successful percentage of the neighbourhood structure applied to random neighbourhood selection. The highest successful percentage is set to be the first neighbourhood structure in the sequence followed by the least successful neighbourhood structure and so on.
3. "**General**" sequence of neighbourhood selection: This sequence is obtained from the best sequence for each tested dataset, and then a general sequence is formed.

Three types of experiments were conducted in this work to investigate the impact of the different sequences of the neighbourhood structures on the quality of the solution obtained. A "Random" sequence is tested based on the neighbourhood structure, which is selected at random without any prior knowledge. A "Best" sequence is implemented based on some knowledge. A "General" sequence is proposed with the purpose of using a sequence of neighbourhood structures that is not dependent on the problem domains or instances. These experiments later help to determine whether prior knowledge is needed before any neighbourhood structures is employed within the local search or whether any problem domains require a specific list of neighbourhood structures.

According to variable neighbourhood search (see [47]), different neighbourhood structures explore different areas in the search space. Moreover, by using different neighbourhood structures, they help the algorithm to escape from local optima. Furthermore, some neighbourhood structures may work well only at the beginning of a search, while other may perform well towards the end. Thus, the neighbourhood sequence is very important to identify the most suitable one for the current state. However, finding the best generic sequence becomes a problem if it is not instance dependent. Therefore, in this work we utilised the Tabu list to prohibit such a situation during the search process based on the previous history the sequences. The importance of the sequence of a neighbourhood structure is widely investigated in the multi-meme method (which is related to our work) and hyper-heuristics (for an example, see [52,53]).

The following sections illustrate the experiments over the "Random", "Best" and "General" selections of neighbourhood structures on ITC2007 datasets. Note that for each experiment in this work, we ran the algorithm 11 times for each instance.

4.1. Random and best selections neighbourhood structures for curriculum-based course timetabling ITC2007-Track 3

The first series of experiments in this section deal with the curriculum-based course timetabling ITC2007-Track 3. Table 9 provides a summary of the results achieved by the Tabu-based memetic algorithm with the "Random" selection of neighbourhood structures. The results indicate that the algorithm is able to enhance the quality of the initial solutions and able to obtain seven best solutions (presented in bold). Note that the time reported here is the amount of time spent to achieve the best solution.

The statistical details of the successful neighbourhood structures used are reported in Table 10. These results lead us to conclude that some neighbourhood structures are able to enhance the quality of solutions to a greater extent than others.

Fig. 8 shows the behaviour of our algorithm when exploring the search space using the *comp01* and *comp12* datasets, respectively. The *x*-axis represents the generation steps, while the *y*-axis represents the penalty cost. The distribution points in these diagrams show the correlation between the number of generations and the solution quality. An analysis of the diagrams shows that the cost improves as the number of generation increases. The slope of the curves is relatively steep, which indicates the great improvement in the quality of solutions at the beginning of the search in all figures where there is possibly much of room for improvement. The degree of improvement becomes relatively lower as the number of generations increases.

From the first experiment with a "Random" selection of neighbourhood structures, we identify the frequency of successful neighbourhood structures that are able to improve the quality of the solution as presented in Table 10. Note that the frequency is calculated by taking into account all the executions until the termination criterion is met. From this ranking, we form a sequence called the "Best" sequence neighbourhood structures as shown in Fig. 9, where Nbs_2 represents the highest percentage and Nbs_5 represents the lowest percentage.

Table 11 reports the results of applying the "Best" sequence to ITC2007-Track 3; this sequence is able to obtain better results compared to the "Random" sequence of neighbourhood structures. Once the neighbourhood structure is unable to produce good results, the next neighbourhood structures in the "Best" sequence will be used. By using the "Best" sequence, the algorithm is able to obtain eight best results (ties with best known results).

4.2. Random and best selections of neighbourhood structures for examination timetabling problems ITC2007-Track 1

We used the same parameter settings as those listed in Table 8. Again, the termination criterion was set to 600 s. The algorithm was run 11 times on each dataset with different random seeds. Table 12 shows the best and average solutions obtained.

Table 9
Results of applying Tabu-based memetic algorithm to curriculum-based course timetabling ITC2007-Track 3 using the "Random" neighbourhood structures.

Instance	Our approach			Best known results[a]	
	Best	Average	Time (s)	Results	Method used
comp01	**5**	5.00	319	5	Tabu Search
comp02	30	42.72	321	24	SAT-Modulo-Theory
comp03	70	78.36	429	66	Local Search
comp04	**35**	51.91	593	35	Local Search
comp05	300	327.36	394	292	Local Search
comp06	42	63.55	471	27	SAT-Modulo-Theory
comp07	8	13.64	505	6	SAT-Modulo-Theory
comp08	**37**	39.45	209	37	Hybridised Algorithm
comp09	100	106.27	458	96	Tabu Search
comp10	7	10.91	427	4	SAT-Modulo-Theory
comp11	**0**	0.00	138	0	Tabu Search
comp12	323	332.45	384	310	Tabu Search
comp13	**59**	63.55	296	59	Tabu Search
comp14	55	69.27	479	51	Mathematical Programming
comp15	70	77.55	592	66	Tabu Search
comp16	**18**	30.55	410	18	SAT-Modulo-Theory
comp17	65	83.91	142	60	SAT-Modulo-Theory
comp18	72	80.18	502	65	Tabu Search
comp19	58	69.82	214	57	Local Search
comp20	11	17.36	496	4	SAT-Modulo-Theory
comp21	**86**	111.36	300	86	SAT-Modulo-Theory

[a] Taken from http://tabu.diegm.uniud.it/ctt/.

Table 10
Percentage of successful neighbourhood structures applied to curriculum-based course timetabling ITC2007-Track 3.

Neighbourhood	Successful %	Rank
Nbs_1	9.67	4
Nbs_2	41.14	1
Nbs_3	17.45	2
Nbs_4	9.30	5
Nbs_5	0.95	9
Nbs_6	2.75	7
Nbs_7	2.12	8
Nbs_8	4.73	6
Nbs_9	11.82	3

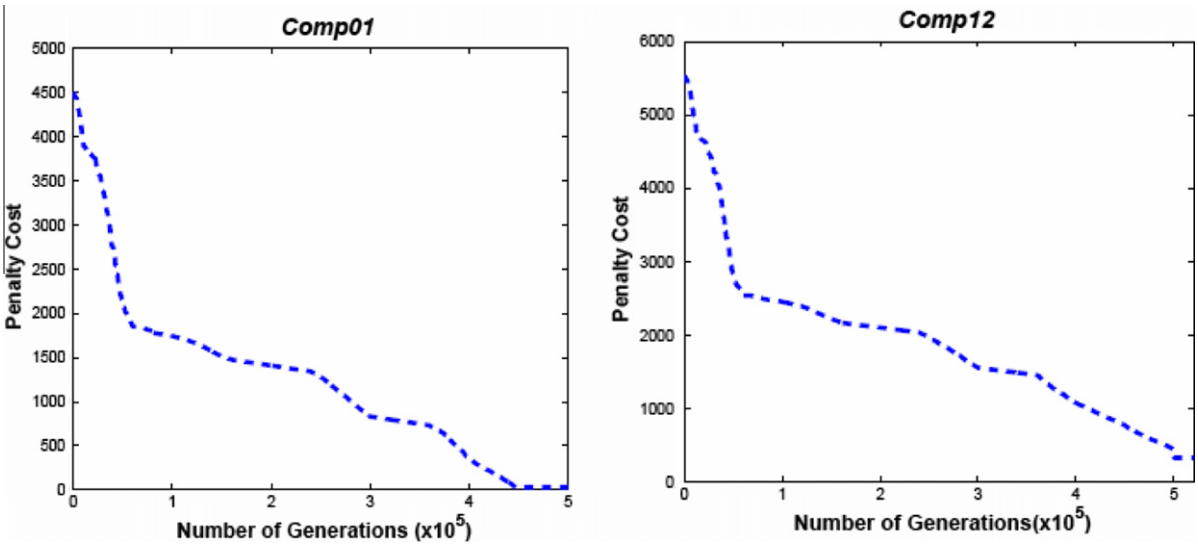

Fig. 8. Convergence of applying a "Random" sequence Tabu-based memetic algorithm to comp01 and comp12 datasets.

Nbs_2	Nbs_3	Nbs_9	Nbs_1	Nbs_4	Nbs_8	Nbs_6	Nbs_7	Nbs_5

Fig. 9. Best sequence of neighbourhood structures for curriculum-based course timetabling ITC2007-Track 3.

Table 11
Results of applying Tabu-based memetic algorithm to curriculum-based course timetabling ITC2007-Track 3 using the "Best" neighbourhood structures.

Instance	Our approach			Best known results[a]	
	Best	Average	Time (s)	Results	Method used
comp01	**5**	5.00	300	5	Tabu Search
comp02	26	36.36	289	24	SAT-Modulo-Theory
comp03	70	74.36	490	66	Local Search
comp04	**35**	38.45	432	35	Local Search
comp05	295	314.45	348	292	Local Search
comp06	30	45.27	320	27	SAT-Modulo-Theory
comp07	7	12.00	532	6	SAT-Modulo-Theory
comp08	**37**	40.82	123	37	Hybridised Algorithm
comp09	102	108.36	234	96	Tabu Search
comp10	5	8.36	342	4	SAT-Modulo-Theory
comp11	**0**	0.00	396	0	Tabu Search
comp12	315	320.27	309	310	Tabu Search
comp13	**59**	64.27	401	59	Tabu Search
comp14	61	64.36	345	51	Mathematical Programming
comp15	69	72.73	356	66	Tabu Search
comp16	**18**	23.73	349	18	SAT-Modulo-Theory
comp17	60	76.36	401	60	SAT-Modulo-Theory
comp18	69	75.64	461	65	Tabu Search
comp19	**57**	66.82	409	57	Local Search
comp20	7	13.45	167	4	SAT-Modulo-Theory
comp21	**86**	100.73	358	86	SAT-Modulo-Theory

[a] Taken from http://tabu.diegm.uniud.it/ctt/.

Fig. 10 shows the behaviour of our algorithm when exploring the search space using the *Exam1*, *Exam2*, *Exam4* and *Exam7* datasets, respectively. Again, the *x*-axis represents the number of generations, while the *y*-axis represents the penalty cost. They show that the penalty cost can be quickly reduced at the beginning of the search, when there is possibly much room for improvement, especially for *Exam1*, *Exam2* and *Exam7*. For *Exam4*, the trend of the graph is slightly different: it is difficult to produce an improvement during the early stages of the search process. However, improvement gradually occurs throughout the search process. The "Random" sequence is able to obtain two best results.

Table 12
Results of applying Tabu-based memetic algorithm to examination timetabling ITC2007-Track 1 using the "Random" neighbourhood structures.

Instances	Best	Average	Best known results
Exam1	**4321**	4490.27	4370 [60]
Exam2	**385**	410.45	385 [46]
Exam3	9942	10530.64	8500 [46]
Exam4	17,494	18590.64	14,879 [46]
Exam5	3035	3490.73	2795 [46]
Exam6	26,010	27145.73	25,410 [46]
Exam7	4117	4590.91	3884 [46]
Exam8	7531	7948.64	7440 [46]

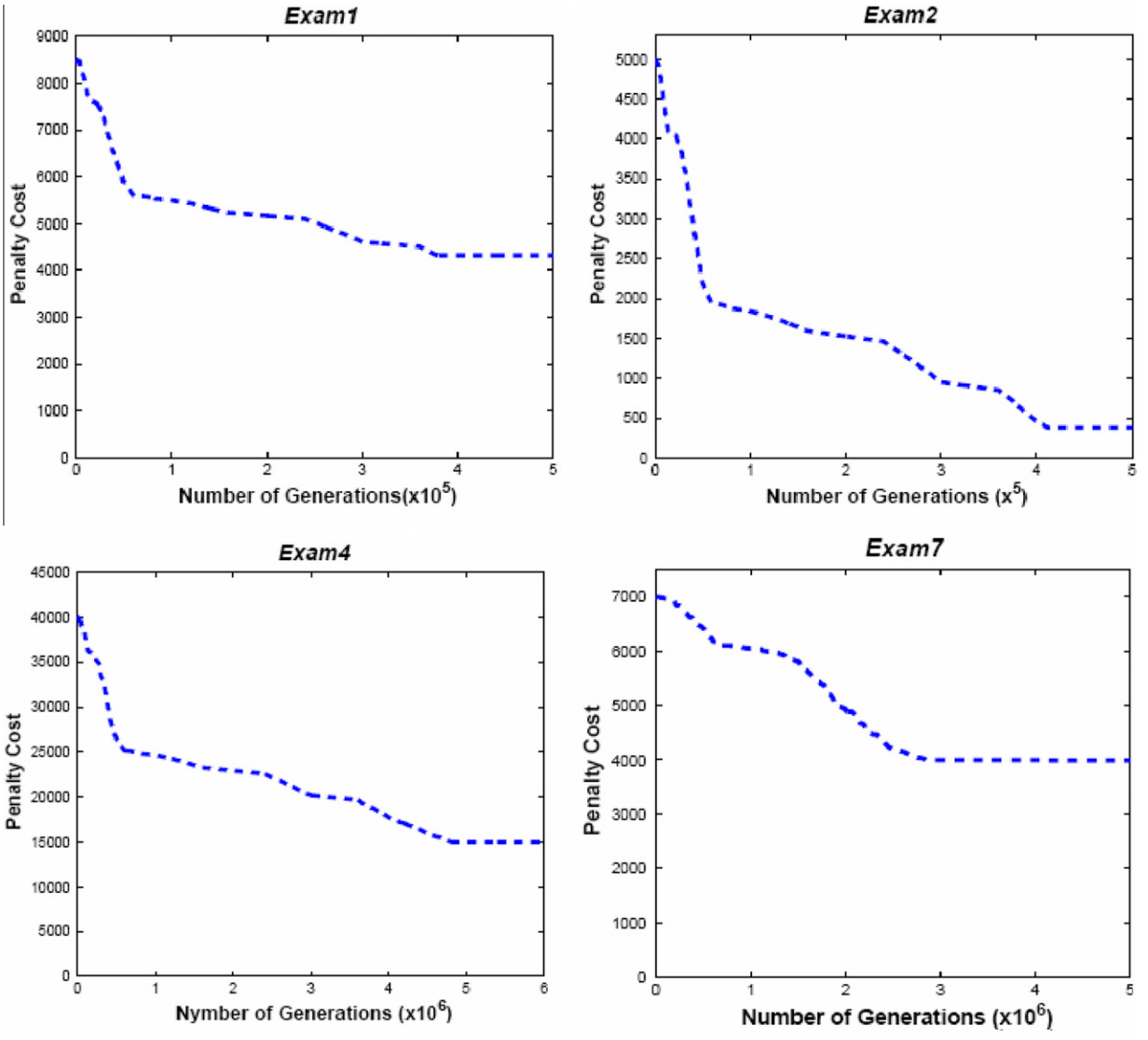

Fig. 10. Convergence of applying a "Random" sequence Tabu-based memetic algorithm to Exam1, Exam2, Exam4 and Exam7 datasets.

Again, we identify the frequency of successful neighbourhood structures that are able to improve the quality of the solution at the time that they are employed, as presented in Table 13.

Based on Table 13, the "Best" sequence of neighbourhood structures, as shown in Fig. 11, is formed, where Nbs_1 represents the highest percentage and Nbs_4 represents the lowest percentage.

Table 13
Percentage of successful applied neighbourhood structures on examination timetabling ITC2007-Track 1.

Neighbourhood	Successful %	Rank
Nbs_1	27.49	1
Nbs_2	26.90	2
Nbs_3	13.82	3
Nbs_4	1.77	9
Nbs_5	4.74	7
Nbs_6	7.29	6
Nbs_7	1.84	8
Nbs_8	8.65	4
Nbs_9	7.45	5

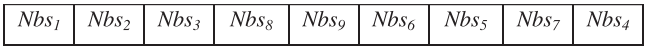

Fig. 11. Best sequence of neighbourhood structures for examination timetabling problem ITC2007-Track 1.

Table 14
Results of applying Tabu-based memetic algorithm to examination timetabling ITC2007-Track 1 using the "Best" neighbourhood structures.

Instances	Best	Average	Best known results
Exam1	**4000**	4386.19	4370 [60]
Exam2	**385**	409.18	385 [46]
Exam3	10,190	10451.45	8500 [46]
Exam4	17,494	18451.45	14,879 [46]
Exam5	3030	3252.73	2795 [46]
Exam6	25,995	26329.45	25,410 [46]
Exam7	4117	4365.18	3884 [46]
Exam8	**7360**	7750.09	7440 [46]

Table 14 shows the results obtained from the execution of the "Best" sequence neighbourhood structures. The results show the large difference between the best and the average value, which indicates that the solutions obtained are not close to each other (i.e., scattered in the solution space). The "Best" sequence is also able to obtain three best solutions, i.e., on *Exam1*, *Exam2* and *Exam8*. However, the qualities of *Exam1* and *Exam8* from the "Best" sequence are better than the qualities of *Exam1* and *Exam8* from the "Random" sequence.

4.3. General selection of neighbourhood structures

Neighbourhood search is one of the general strategies used in designing heuristic algorithms for university timetabling problems. Some algorithms have been proposed to overcome the greatest shortcoming of neighbourhood search, i.e., a tendency to become stuck in a local optimum. Moreover, a neighbourhood performance is not stable over different problem domains. Therefore, in this section, we propose a "General" sequence of neighbourhood structures that is based on our previous experiments with "Random" and "Best" sequences of neighbourhood structures tested on university timetabling problems.

Table 15 shows the "Best" successful neighbourhood structures and the rank for each neighbourhood (taken from previous experiments described in Sections 4.1 and 4.2). We calculate the summation of all the rank for each neighbourhood structure from two categories of problems. For example, Nbs_2 with total rank (denoted as total in Table 15) is 3 at rank 1. The smaller summation value indicates a better neighbourhood structure. From these summations, we form a "General" sequence of neighbourhood structures as shown in Fig. 12, where Nbs_2 represents the highest rank and Nbs_5 represents the lowest rank.

Tables 16 and 17 report the results of applying the "General" sequence of neighbourhood structures to ITC2007-Track 3 and ITC2007-Track 1, respectively. It can be seen that the "General" sequence is able to produce sufficiently good results for both cases.

5. Results analysis

In this section, we perform two types of analysis. The first is based on the total summation of penalties, while the second is based on the *p*-value. From Table 18, we can see that the "Best" sequence outperforms the performance of the "Random" and "General" sequences based on the summation of the total penalty cost for each sequence when tested on curriculum-based

Table 15
The new rank of neighbourhood structures.

Neighbourhood	ITC2007-Track 3	ITC2007-Track 1	Total	Rank
Nbs_1	4	1	5	2
Nbs_2	1	2	3	1
Nbs_3	2	3	5	2
Nbs_4	5	9	14	6
Nbs_5	9	7	16	7
Nbs_6	7	6	13	5
Nbs_7	8	8	16	7
Nbs_8	6	4	10	4
Nbs_9	3	5	8	3

Nbs_2	Nbs_1	Nbs_3	Nbs_9	Nbs_8	Nbs_6	Nbs_4	Nbs_5	Nbs_7

Fig. 12. General sequence of neighbourhood structures.

Table 16
Results of applying Tabu-based memetic algorithm to curriculum-based course timetabling ITC2007-Track 3 using the "General" neighbourhood structures.

Instance	Our approach			Best known results[a]	
	Best	Average	Time (s)	Best	Method used
comp01	**5**	5.00	319	5	Tabu Search
comp02	27	40.45	420	24	SAT-Modulo-Theory
comp03	73	80.45	321	66	Local Search
comp04	39	46.64	390	35	Local Search
comp05	312	320.64	349	292	Local Search
comp06	30	42.64	361	27	SAT-Modulo-Theory
comp07	10	13.45	371	6	SAT-Modulo-Theory
comp08	**37**	40.55	238	37	Hybridised Algorithm
comp09	100	116.36	361	96	Tabu Search
comp10	5	10.45	604	4	SAT-Modulo-Theory
comp11	**0**	0.00	345	0	Tabu Search
comp12	330	339.27	315	310	Tabu Search
comp13	62	64.64	451	59	Tabu Search
comp14	53	65.73	237	51	Mathematical Programming
comp15	73	78.36	361	66	Tabu Search
comp16	18	24.64	451	18	SAT-Modulo-Theory
comp17	61	77.09	371	60	SAT-Modulo-Theory
comp18	79	85.64	445	65	Tabu Search
comp19	**57**	59.55	349	57	Local Search
comp20	**4**	10.45	222	4	SAT-Modulo-Theory
comp21	90	100.00	321	86	SAT-Modulo-Theory

[a] Taken from http://tabu.diegm.uniud.it/ctt/.

Table 17
Results of applying Tabu-based memetic algorithm to examination timetabling ITC2007-Track 1 using the "General" neighbourhood structures.

Instances	Best	Average	Best known results
Exam1	**4350**	4379.64	4370 [60]
Exam2	**385**	410.09	**385** [46]
Exam3	9951	10230.18	**8500** [46]
Exam4	18,000	18330.18	**14,879** [46]
Exam5	3040	3269.73	**2795** [46]
Exam6	26,010	26231.27	**25,410** [46]
Exam7	4250	4481.27	**3884** [46]
Exam8	7450	7535.55	**7440** [46]

course timetabling problems. It is able to obtain 14 instances that are better than or ties the "Random" and "General" sequences. On the other hand, the "General" sequence is able to obtain 6 instances that are better than or ties the "Best" sequence.

Table 18
Results comparison: based on application of "Random", "Best" and "General" sequence of neighbourhood structures to curriculum-based course timetabling for ITC2007-Track 3.

Instance	Neighbourhood structures			Best known results[a]	
	"Random"	"Best"	"General"	Best	Method used
comp01	**5.00**	**5.00**	**5.00**	5	Tabu Search
comp02	42.73	**36.36**	40.45	24	SAT-Modulo-Theory
comp03	78.36	**74.36**	80.45	66	Local Search
comp04	51.91	**38.45**	46.64	35	Local Search
comp05	327.36	**314.45**	320.64	292	Local Search
comp06	63.55	45.27	**42.64**	27	SAT-Modulo-Theory
comp07	13.64	**12.00**	13.45	6	SAT-Modulo-Theory
comp08	**39.45**	40.82	40.55	37	Hybridised Algorithm
comp09	**106.27**	108.36	116.36	96	Tabu Search
comp10	10.91	**8.36**	10.45	4	SAT-Modulo-Theory
comp11	**0.00**	**0.00**	**0.00**	0	Tabu Search
comp12	332.45	**320.27**	339.27	310	Tabu Search
comp13	**63.55**	64.27	64.64	59	Tabu Search
comp14	69.27	**64.36**	65.73	51	Mathematical Programming
comp15	77.55	**72.73**	78.36	66	Tabu Search
comp16	30.55	**23.73**	24.64	18	SAT-Modulo-Theory
comp17	83.91	**76.36**	77.09	60	SAT-Modulo-Theory
comp18	80.18	**75.64**	85.64	65	Tabu Search
comp19	69.82	66.82	**59.55**	57	Local Search
comp20	17.36	13.45	**10.45**	4	SAT-Modulo-Theory
comp21	111.36	100.73	**100.00**	86	SAT-Modulo-Theory
Total	1675.18	**1561.82**	1622.00		

[a] Taken from http://tabu.diegm.uniud.it/ctt/.

Table 19
Results comparison: based on application of "Random", "Best" and "General" sequence of neighbourhood structures to examination timetabling for ITC2007-Track 1.

Instances	Neighbourhood structures			Best known results
	"Random"	"Best"	"General"	
Exam1	4490.27	4386.18	**4379.64**	4370 [60]
Exam2	410.45	**409.18**	410.09	385 [46]
Exam3	10530.64	10451.45	**10230.18**	8500 [46]
Exam4	18590.64	18451.45	**18330.18**	14,879 [46]
Exam5	3490.73	**3252.73**	3269.73	2795 [46]
Exam6	27145.73	26329.45	**26231.27**	25,410 [46]
Exam7	4590.91	**4365.18**	4481.27	3884 [46]
Exam8	7948.64	7750.09	**7535.55**	7440 [46]
Total	77198.00	75397.73	**74867.91**	

In Table 19, the "General" sequence outperforms the performance of the "Best" and "Random" sequences when tested on examination timetabling problems. The results show that the "General" sequence is able to obtain five best results (represented in bold). The results also show that the "General" sequence works best, followed by the "Best" sequence and "Random" sequence, successively. Based on these results, where no sequence is able to obtain the best results for all instances, we believe that in solving different problem domains we may need a tailor-made sequence of neighbourhood structures to obtain high-quality solutions. This requires preliminary experiments to obtain the correct sequence of neighbourhood structures. We can see that, for example, *Nbs1* works well for curriculum-based course timetabling problems (1st rank) but not for the examination timetabling problem (4th rank). However, there are a certain cases in which the neighbourhood structure performs quiet similarly for both problems, i.e., *Nbs2* (1st rank on ITC2007-Track 3, and 2nd rank on ITC2007-Track 1), *Nbs3* (2nd rank on ITC2007-Track 3, and 3rd rank on ITC2007-Track 1), and *Nbs8* (8th rank on both tracks). This shows that different problems might require a different sequence of neighbourhood structures to be employed during the search process while searching for better solutions to the problem at hand.

Since the results reveal that the best sequence of neighbourhood structures that is used for the curriculum-based course timetabling problem (i.e. "Best") is different from the one that is employed for the examination timetabling problem (i.e. "General"), thus we execute a statistical test to examine if there is any significant difference between sequences with the critical level = 0.05 (see [11]). We execute a pair comparison as follow:

Table 20
p-Value results for curriculum-based course timetabling for ITC2007-Track 3.

Instance	*p*-Value	
	(B–R)	(B–G)
comp01	1	1
comp02	**0.0401**	**0.0198**
comp03	0.0570	**0.0013**
comp04	**0.0021**	**0.0002**
comp05	0.1335	**0.0026**
comp06	**0.0038**	0.2058
comp07	0.4518	**0.0105**
comp08	0.2381	0.2711
comp09	0.1566	**0.0051**
comp10	**0.0416**	0.0573
comp11	1	1
comp12	**1.81E−05**	**4.34E−08**
comp13	0.2136	0.2503
comp14	**0.0401**	0.3192
comp15	**0.0044**	**0.0017**
comp16	0.0545	0.1758
comp17	**0.0340**	0.2531
comp18	**0.0076**	**0.0003**
comp19	0.1003	**0.0203**
comp20	**0.0039**	0.3178
comp21	**0.0229**	0.2271

Table 21
p-Value results for examination timetabling for ITC2007-Track 1.

Instance	*p*-Value	
	(G–R)	(G–B)
Exam1	**0.0467**	0.2613
Exam2	0.4605	0.1602
Exam3	0.3836	**0.0270**
Exam4	**0.0117**	**0.0467**
Exam5	0.0917	0.0953
Exam6	**0.0012**	0.1182
Exam7	0.2417	**0.0036**
Exam8	**3.36E−5**	**0.0194**

- "Best vs. Random (B–R)" and "Best vs. General (B–G)" sequences for the curriculum-based course timetabling ITC2007-Track 3
- "General vs. Random (G–R)" and "General vs. Best (G–B)" sequences for the examination timetabling ITC2007-Track 1

Table 20 represents the *p*-value for the curriculum-based course timetabling ITC2007-Track 3. We can see that the "Best" sequence outperforms the "Random" and "General" sequences on 11 and 10 out of 21 instances, which are about 52.4% and 47.6%, respectively. Table 21 shows the *p*-value for the examination timetabling problem ITC2007-Track 1 where the "General" sequence outperforms both "Random" and "Best" sequences on 4 out of 8 instances (which is about 50%). The presented *p*-value lead us to conclude that different sequence of neighbourhood structures has different impact, where the neighbourhood sequence used in the curriculum-based course timetabling and examination timetabling problems are different from each other. Thus, different problems might need different sequences of neighbourhood structures to be employed during the search process.

We further validate our results and determine whether there are significant differences between the proposed algorithm and the best-performing algorithms of the benchmark by conducting a statistical analysis using the average results for each instance of both problems. Note that in this analysis, we compare our results with those of five winners of the International Timetabling Competition 2007, which can be accessed at http://www.cs.qub.ac.uk/itc2007/.

Friedman's test is carried out to determine whether there are significant differences between our method and the best-performance approaches. If significant differences are detected (based on Friedman's test), Holm's and Hochberg's tests are conducted as post hoc methods to obtain the adjusted *p*-values for each comparison between the control algorithm (the best-performing one in the comparison, the one has 1st rank according to Friedman's test as shown for example in Table 23) and the rest of algorithms performed (see [44,45]).

ITC2007-Track 1

The winners of ITC2007-Track 1 are (taken from http://www.cs.qub.ac.uk/itc2007/)

1st Place: Tomas Müller
2nd Place: Christos Gogos
3rd Place: Mitsunori Atsuta, Koji Nonobe, and Toshihide Ibaraki
4th Place: Geoffrey De Smet
5th Place: Nelishia Pillay

The average results for ITC2007-Track 1 are given in Table 22. Note that these results are compiled from http://www.cs.qub.ac.uk/itc2007/.

Table 23 summarises the ranking obtained by Friedman's test. Note that Geoffrey De Smet's results are not considered in the Friedman test because the author was unable to obtain any feasible solutions on *Exam3*, *Exam4* and *Exam8* datasets (denoted by '-').

Table 23 shows that the Tabu-based memetic algorithm is ranked first, followed by Muller, Gogos, Atsuta and Pillay, successively. The *p*-value computed by Friedman's test is 5.3754E−6, which is below the critical level ($\alpha = 0.05$). This value shows that there are significant differences among the observed results. Therefore, we conducted post hoc methods (i.e., Holm's and Hochberg's test) for the control Tabu-based memetic algorithm as well as other methods (see [44,45]). Table 24 shows the adjusted *p*-values (Friedman).

Holm's and Hockberg's procedures reveal significant differences when using the Tabu-based memetic algorithm as a control algorithm, where Tabu-based memetic algorithm is better than Pillay's algorithm, Atsuta's algorithm and Gogos's algorithm, with $\alpha = 0.05$ (3/4 algorithms). In fact, both Holm's and Hochberg's procedures confirm that Muller's algorithm is the best in the competition. However, even though Muller's algorithm is superior to the rest of the algorithms, the average results reported in Table 22 show that our approach outperforms Muller's in 7 out of 8 datasets. This shows that the performance of our approach is close to Muller's algorithm (if not better in some instances).

Table 22
Average results for ITC2007-Track 1.

Instances	Tabu-based memetic	Muller	Gogos	Atsuta	De Smet	Pillay
Exam1	4379.64	4574.90	6064.00	9083.90	6670.80	12819.20
Exam2	410.09	414.00	1048.60	3669.40	623.00	3925.80
Exam3	10230.18	10789.17	14133.50	19367.40	–	19812.10
Exam4	18330.18	21639.00	20666.60	26346.80	–	25728.80
Exam5	3269.73	3320.70	4229.10	4920.30	3858.40	11176.00
Exam6	26231.27	27808.50	28077.50	29935.00	28155.00	34028.89
Exam7	4481.27	4396.30	6759.50	11004.33	5432.30	19669.30
Exam8	7535.55	7950.30	10809.00	14869.90	–	16720.70

Table 23
Average rankings of the algorithms (Friedman) used for ITC2007-Track 1.

Algorithm	Ranking
Tabu-based memetic algorithm	1.125
Muller	2.000
Gogos	2.875
Atsuta	4.125
Pillay	4.875

Table 24
Adjusted *p*-values (Friedman) on ITC2007-Track 1.

	Algorithm	Unadjusted	PHolm	PHoch
1	Pillay	2.1014E−6	8.4057E−6	8.4057E−6
2	Atsuta	1.4780E−4	4.4341E−4	4.4341E−4
3	Gogos	0.0269	0.0537	0.0537
4	Muller	0.2684	0.2684	0.2684

ITC2007-Track 3

The winners of ITC2007-Track 3 are (taken from http://www.cs.qub.ac.uk/itc2007/)

1st Place: Tomas Muller
2nd Place: Zhipeng Lu and Jin-Kao Hao
3rd Place: Mitsunori Atsuta, Koji Nonobe, and Toshihide Ibaraki
4th Place: Martin Josef Geiger
5th Place: Michael Clark, Martin Henz, and Bruce Love

The average results for ITC2007-Track 3 are given in Table 25. Note that these results are compiled from http://www.cs.qub.ac.uk/itc2007/.

Similar to the treatment of the previous dataset, we apply Friedman's procedure to determine whether there are significant differences in the results shown in Table 25. The p-value computed by Friedman's test is 5.4327E−11, which is again below the level of significance $\alpha = 0.05$. As a result, there are significant differences among the observed results. Table 26 summarises the rankings obtained by Friedman's test.

We further conducted Holm's and Hochberg's procedure to compare the control algorithm (i.e., Tabu-based memetic algorithm) with the rest of the algorithms by comparing p-values. Table 27 shows the adjusted p-values obtained by Holm's and Hockberg's procedures for ITC2007-Track 3.

Table 25
Average results for ITC2007-Track 3.

Instances	Tabu-based memetic	Muller	LuHoa	Atsuta	Geiger	Clark
comp01	5.00	5.00	5.00	5.10	6.70	27.00
comp02	36.36	61.30	61.20	65.60	142.70	131.10
comp03	74.36	94.80	84.50	89.10	160.30	138.40
comp04	38.45	42.80	46.90	39.20	82.00	90.20
comp05	314.45	343.50	326.00	334.50	525.40	811.50
comp06	45.27	56.80	69.40	74.10	110.80	149.30
comp07	12.00	33.90	41.50	49.80	76.60	153.4
comp08	40.82	46.50	52.60	46.00	81.70	96.50
comp09	108.36	113.10	116.50	113.30	164.10	148.90
comp10	8.36	21.30	34.80	36.90	81.30	101.30
comp11	0.00	0.00	0.00	0.00	0.30	5.70
comp12	320.27	351.60	360.10	361.60	485.10	445.30
comp13	64.27	73.90	79.20	76.10	110.40	122.90
comp14	64.36	61.80	65.90	62.30	99.00	105.90
comp15	72.73	94.80	84.50	89.10	160.30	138.00
comp16	23.73	41.20	49.10	50.20	92.60	107.30
comp17	76.36	86.60	100.70	107.30	143.40	166.60
comp18	75.64	91.70	80.70	73.30	129.40	126.80
comp19	66.82	68.80	69.50	79.60	132.80	125.40
comp20	13.45	34.30	60.90	65.00	97.50	179.30
comp21	100.73	108.00	124.70	138.10	185.30	185.80

Table 26
Average rankings of the algorithms (Friedman) for ITC2007-Track 3.

Algorithm	Ranking
Tabu-based memetic algorithm	1.3571
Muller	2.5952
LuHoa	3.0714
Atsuta	3.3571
Geiger	5.1429
Clark	5.4762

Table 27
Adjusted p-values (Friedman) for ITC2007-Track 3.

	Algorithm	Unadjusted	PHolm	PHoch
1	Clark	9.7210E−13	4.8605E−12	4.8605E−12
2	Geiger	5.4883E−11	2.1953E−10	2.1953E−10
3	Atsuta	5.3201E−4	0.0016	0.0016
4	LuHoa	0.0030	0.0060	0.0060
5	Muller	0.0320	0.0320	0.0320

Holm's and Hockberg's procedures show significant differences in the use of the Tabu-based memetic algorithm as the control algorithm. Tabu-based memetic algorithm is better than all of the algorithms, with $\alpha = 0.05$ (5/5 algorithms). The performance reported from the statistical test is also supported by the average results shown in Table 25, where the Tabu-based memetic algorithm is better than the first-ranked winner (i.e., Muller's algorithm) in 18 out of 21 datasets (ties on *comp01* and *comp11* datasets).

6. Conclusion

The overall goal of the work presented in this paper is to investigate the performance of the sequence of neighbourhood structures in addressing university timetabling problems. The approach is tested on two university timetabling benchmark datasets. This approach is simple yet effective and manages to produce a number of best results. It is shown to be good across all of the benchmark problems in comparison with other approaches studied in the literature. Finding the correct sequence of neighbourhood structures is one of the main contributions of this work. Starting with the experiment on a "Random" selection of the neighbourhood structures, we form a "Best" sequence of neighbourhood structures for each case study. Then, a "General" sequence is formed and tested on the same domains. The purpose here is to use a sequence of neighbourhood structures that is independent of the problem domains. Furthermore, we discovered that only *Nbs2* exhibits better performance on both tested domains. This indicates that different problems might require different sequences of neighbourhood structures to be employed during the search process to obtain high-quality solutions for a given problem.

Acknowledgements

We are very grateful for the support and valuable discussion given by Nasser R. Sabar. We are also very grateful to the anonymous referees, whose thoughtful and considered comments significantly improved the paper.

References

[1] S. Abdullah, S. Ahmadi, E.K. Burke, M. Dror, Investigating Ahuja-Orlin's large neighbourhood search approach for examination timetabling, OR Spectrum 29 (2) (2007) 351–372.
[2] S. Abdullah, S. Ahmadi, E.K. Burke, M. Dror, B. McCollum, A tabu based large neighbourhood search methodology for capacitated examination timetabling problem, Journal of Operational Research 58 (2007) 494–502.
[3] S. Abdullah, E.K. Burke, B. McCollum, in: K.F. Doerner, M. Gendreau, P. Greistorfer, W.J. Gutjahr, R.F. Hartl, M. Reimann (Eds.), Metaheuristics – Progress in Complex Systems Optimization, Computer Science Interfaces Book Series, vol. 39, Springer Operations Research, 2007, pp. 153–169. ISBN-13:978-0-387-71919-1.
[4] S. Abdullah, E.K. Burke, B. McCollum, A hybrid evolutionary approach to the university course timetabling problem, in: IEEE Congress on Evolutionary Computation, Singapore, 25–28 September, 2007, pp. 1764–1768, ISBN: 1-4244-1340-0.
[5] S. Abdullah, H. Turabieh, Generating university course timetable using genetic algorithm and local search, in: Proceeding of the 3rd International Conference on Hybrid Information Technology, 2008, pp. 254–260.
[6] S. Abdullah, H. Turabieh, B. McCollum, Electromagnetism-like mechanism with force decay rate great deluge for the course timetabling problem, Lecture Notes in Computer Science, vol. 5589, Springer, 2009, pp. 497–504.
[7] S. Abdullah, H. Turabieh, B. McCollum, P. McMullan, A hybrid metaheuristic approach to the university course timetabling problem, Journal of Heuristics (2010), doi:10.1007/s10732-010-9154-y.
[8] S. Abdullah, K. Shaker, B. McCollum, P. McMullan, Incorporating great deluge with kempe chain neighbourhood structure for the enrolment-based course timetabling problem, in: J. Yu et al. (Eds.), RSKT 2010, LNAI, vol. 6401, Springer-Verlag, Berlin Heidelberg, 2010, pp. 70–77.
[9] S. Abdullah, K. Shaker, B. McCollum, P. McMullan, Dual sequence simulated annealing with round-robin approach for university course timetabling problems, in: EvoCop 2010, Lecture Notes in Computer Science, vol. 6022/2010, Springer, 2010, pp. 1–10.
[10] M. Abramowitz, I.A. Stegun, Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, Ninth dover printing, Tenth gpo printing ed., Dover, New York, 1964.
[11] A.A. Afifi, S.P. Azen, Statistical Analysis: A Computer Oriented Approach, Academic Press, Inc., Orlando, FL, USA, 1979.
[12] S. Ahmadi, R. Barone, P. Cheng, P. Cowling, B. McCollum, Perturbation based variable neighbourhood search in heuristic space for examination timetabling problem, in: Proceedings of the 1st Multidisciplinary International Conference on Scheduling: Theory and Applications, 2003, pp. 155–171.
[13] H. Asmuni, E.K. Burke, J. Garibaldi, B. McCollum, Fuzzy multiple ordering criteria for examination timetabling, in: E.K. Burke, M. Trick (Eds.), Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 3616, Springer, 2005, pp. 334–353.
[14] H. Asmuni, E.K. Burke, J. Garibaldi, B. McCollum, A novel fuzzy approach to evaluate the quality of examination timetabling, in: E.K. Burke, H. Rudova (Eds.), Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 3867, Springer, 2007, pp. 327–346.
[15] B. Bilgin, E. Ozcan, E.E. Korkmaz, An experimental study on hyper-heuristics on exam timetabling, in: E.K. Burke, H. Rudova (Eds.), Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 3867, Springer, 2007, pp. 394–412.
[16] P. Boizumault, Y. Delon, L. Peridy, Constraint logic programming for examination timetabling, The Journal of Logic Programming 26 (1996) 217–233.
[17] D. Brelaz, New methods to colour the vertices of a graph, Communication of ACM 22 (1979) 251–256.
[18] E.K. Burke, Y. Bykov, J.P. Newall, S. Petrovic, A time-predefined local search approach to exam timetabling problem, IIE Transactions 36 (6) (2004) 509–528.
[19] E.K. Burke, J.P. Newall, R.F. Weare, A memetic algorithm for university exam timetabling, in: E.K. Burke, P. Ross (Eds.), Selected Papers from 1st International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 1153, Springer, 1996, pp. 241–250.
[20] E.K. Burke, J.P. Newall, R.F. Weare, Initialisation strategies and diversity in evolutionary timetabling, Evolutionary Computation 6 (1) (1998) 81–103.
[21] E.K. Burke, M. Dror, S. Petrovic, R. Qu, Hybrid graph heuristics in hyper-heuristic applied to exam timetabling, in: B.L. Golden, S. Raghavan, E.A. Wasil (Eds.), The Next Wave in Computing, Optimisation, and Decision Technologies, Springer, 2005, pp. 79–91.
[22] E.K. Burke, S. Petrovic, R. Qu, Case-based heuristic selection for timetabling problems, Journal of Scheduling 9 (2006) 115–132.

[23] E.K. Burke, B. McCollum, A. Meisels, S. Petrovic, R. Qu, A graph based hyper-heuristic for exam timetabling, European Journal of Operational Research 176 (2007) 177–192.
[24] E.K. Burke, G. Kendall, E. Soubeiga, A tabu-search hyper-heuristic for timetabling and rostering, Journal of Heuristics 9 (6) (2003) 451–470.
[25] E.K. Burke, A. Eckersley, B. McCollum, S. Petrovic, R. Qu, Hybrid variable neighbourhood approaches to university exam timetabling, European Journal of Operational Research 203 (2) (2010) 484–493.
[26] E.K. Burke, J.P. Newall, A multistage evolutionary algorithm for the timetable problem, IEEE Transactions on Evolutionary Computation 3 (1) (1999) 63–74.
[27] E.K. Burke, J.P. Newall, Solving examination timetabling problems through adaption of heuristic orderings, Annals OR 129 (1–4) (2004) 107–134.
[28] E.K. Burke, J. Kingston, D. de Werra, Applications to timetabling, in: J. Gross, J. Yellen (Eds.), Handbook of Graph Theory, Chapman Hall/CRC Press, 2004, pp. 445–474.
[29] M. Caramia, P. Dell'Olmo, G.F. Italiano, New algorithms for examination timetabling, in: Algorithms Engineering 4th International Workshop, Proceedings WAE 2001, Lecture Notes in Computer Science, vol. 1982, Springer, Saarbrücken, Germany, 2001, pp. 230–241.
[30] M.W. Carter, A survey of practical applications of examination timetabling algorithms, Operations Research 34 (2) (1986) 193–202.
[31] M.W. Carter, D.G. Johnson, Extended clique initialisation in examination timetabling, Journal of Operational Research Society 52 (2001) 538–544.
[32] S. Casey, J. Thompson, GRASPing the examination scheduling problem, in: E.K. Burke, P. De Causmaecker (Eds.), Selected Papers from 4th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 2740, Springer, 2003, pp. 232–244.
[33] C.Y. Cheong, K.C. Tan, B. Veeravalli, A multi-objective evolutionary algorithm for examination timetabling, Journal of Scheduling 12 (2) (2009) 121–146.
[34] D. Corne, P. Ross, H.L. Fang, Fast practical evolutionary timetabling, in: Proceedings of Evolutionary Computing AISB Workshop, Lecture Notes in Computer Science, vol. 865, Springer, 1994, pp. 251–263.
[35] P. Côté, T. Wong, R. Sabourin, A hybrid multi-objective evolutionary algorithm for the uncapacitated exam proximity problem, in: E.K. Burke, M. Trick (Eds.), Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 3616, Springer, 2005, pp. 294–312.
[36] P. David, A constraint-based approach for examination timetabling using local repair technique, in: E.K. Burke, M.W. Carter (Eds.), Selected Papers from the 2nd International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 1408, Springer, 1998, pp. 169–186.
[37] L. Di Gaspero, Recolor, shake and kick: a recipe for the examination timetabling problem, in: The Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling (PATAT IV), Gent, Belgium, 2002, pp. 404–407.
[38] L. Di Gaspero L, A. Schaerf, Tabu search techniques for examination timetabling, in: E.K. Burke, W. Erben (Eds.), Selected Papers from the 3rd International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 2079, Springer, 2001, pp. 104–117.
[39] L. Di Gaspero, B. McCollum, A. Schaerf, The second international timetabling competition (ITC-2007): curriculum-based course timetabling (Track 3), in: International Workshop on Scheduling a Scheduling Competition, 7th International Conference on Automated Planning and Scheduling September 22nd–26th, 2007, Providence, Rhode Island, USA, 2007.
[40] K.A. Dowsland, J.M. Thompson, Ant colony optimisation for the examination scheduling problem, Journal of the Operational Research Society 56 (2005) 426–438.
[41] M. Eley, Ant algorithms for the exam timetabling, in: E.K. Burke, H. Rudova (Eds.), Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 3867, Springer, 2007, pp. 364–382.
[42] W. Erben, A grouping genetic algorithm for graph colouring and exam timetabling, in: E.K. Burke, P. De Causmaecker (Eds.), Selected Papers from 4th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 2740, Springer, 2001, pp. 132–156.
[43] E. Ersoy, E. Ozcan, A.S. Etaner, Memetic algorithms and hyperhill-climbers, in: Proceedings of the 3rd Multidisciplinary International Conference on Scheduling: Theory and Applications, 2007, pp. 159–166.
[44] S. Garcia, D. Molina, M. Lozano, F. Herrera, A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization, Journal of Heuristics 15 (2009) 617–644.
[45] S. Garcia, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power, Information Sciences 180 (10) (2010) 2044–2064.
[46] C. Gogos, G. Goulas, P. Alefragis, E. Housos, Pursuit of better results for the examination timetabling problem using grid resources, in: Proceedings of IEEE Symposium on Computational Intelligence in Scheduling (CI-Sched), 2009, pp. 22–28.
[47] P. Hansen, N. Mladenović, Variable neighbourhood search, European Journal of Operational Research 130 (2001) 449–467.
[48] G. Kendall, N.M. Hussin, An investigation of a tabu search based hyper-heuristic for examination timetabling, in: G. Kendall, E.K. Burke, S. Petrovic (Eds.), Selected Papers from Multidisciplinary Scheduling: Theory and Applications, 2005, pp. 309–328.
[49] R. Lewis, A survey of metaheuristic-based techniques for university timetabling problems, OR Spectrum 30 (1) (2008) 167–190.
[50] S.L.M. Lim, A broker algorithm for timetabling problem, in: E.K. Burke, P.De. Causmaecker (Eds.), Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling, 2002, pp. 372–386.
[51] H.R. Lourenco, O. Martin, T. Stutzle, Iterated local search, in: F. Glover, G.A. Kochenberher (Eds.), Handbook of Metaheuristics, Kluwer Academic, Boston, 2003, pp. 321–354.
[52] Z. Lu, J.K. Hao, Adaptive tabu search for course timetabling, European Journal of Operational Research 200 (1) (2010) 235–244.
[53] Z. Lu, J. Hao, F. Golver, Neighborhood analysis: a case study on curriculum-based course timetabling, Journal of Heuristics 17 (2) (2011) 97–118.
[54] M.R. Malim, A.T. Khader, A. Mustafa, Artificial immune algorithms for university timetabling, in: E.K. Burke, H. Rudova (Eds.), Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 3867, Springer, 2006, pp. 234–245.
[55] B. McCollum, A perspective on bridging the gap between theory and practice in university timetabling, in: E.K. Burke, H. Rudova (Eds.), Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 3867, Springer, 2007, pp. 3–23.
[56] B. McCollum, P. McMullan, A.J. Parkers, E.K. Burke, R. Qu, A new method for automated examination timetabling, Annals of Operations Research, 2011, doi:10.1007/s10479-011-0997-x.
[57] B. McCollum, P. McMullan, E.K. Burke, A.J. Parkes, S. Abdullah, An extended great deluge approach to the examination timetabling problem, in: The 4th Multidisciplinary International Scheduling Conference: Theory and Applications, 2009, pp. 424–434.
[58] L.T.G. Merlot, N. Boland, B.D. Hughes, P.J. Stuckey, A hybrid algorithm for the examination timetabling problem, in: E.K. Burke, P. De Causmaecker (Eds.), Selected Papers from 4th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 2740, Springer, 2003, pp. 207–231.
[59] N. Mladenovic, P. Hansen, Variable neighbourhood search, Computers and Operations Research 24 (11) (1997) 1097–1100.
[60] T. Muller, ITC2007: solver description: a hybrid approach, Annals of Operations Research 27 (1) (2009) 429–446.
[61] Z. NajiAzimi, Comparison of metaheuristic algorithms for examination timetabling problem, Applied Mathematics and Computation 16 (1–2) (2004) 337–354.
[62] S. Petrovic, E.K. Burke, Chapter 45: university timetabling, in: J. Leung (Ed.), Handbook of Scheduling: Algorithms, Models, and Performance Analysis, CRC Press, 2004.

[63] P. Pongcharoen, W. Promtet, P. Yenradee, C. Hicks, Stochastic optimisation timetabling tool for university course scheduling, International Journal of Production Economics 112 (2) (2008) 903–918. Special Section on RFID: Technology, Applications, and Impact on Business Operations.
[64] R. Qu, E.K. Burke, Adaptive decomposition and construction for university timetabling problems, in: Multidisciplinary International Scheduling: Theory and Applications (MISTA'07), 2007, pp. 418–425.
[65] R. Qu, E.K. Burke, Hybridisations within a graph based hyper-heuristic framework for university timetabling problems, Journal of Operational Research Society 60 (2009) 1273–1285.
[66] R. Qu, E.K. Burke, B. McCollum, L.T.G. Merlot, A survey of search methodologies and automated system development for examination timetabling, Journal of Scheduling 12 (2009) 55–89.
[67] P. Ross, D. Corne, H.L. Fang, Improving evolutionary timetabling with delta evolution and directed mutation, in: The Proceedings of the Conference of Parallel Problem Solving in Nature III, Lecture Notes in Computer Science, vol. 866, Springer, 1994, pp. 556–565.
[68] P. Ross, E. Hart, D. Corne, Some observations about GA based timetabling, in: E.K. Burke, M.W. Carter (Eds.), Selected Papers from the 2nd International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 1408, Springer, 1998, pp. 115–129.
[69] P. Ross, J.G. Marin-Blazquez, E. Hart, Hyper-heuristics applied to class and exam timetabling problems, in: Proceedings of the 2004 Congress on Evolutionary Computation (CEC 2004), 2004, pp. 1691–1698.
[70] N.R. Sabar, M. Ayob, G. Kendall, Solving examination timetabling problems using honey-bee mating optimization (ETP-HBMO), in: Proceedings of the 4th Multidisciplinary International Scheduling Conference: Theory and Applications (MISTA2009), 2009, pp. 399–408.
[71] E.-G. Talbi, Metaheuristic: From Design to Implementation, first ed., Wiley, 2009. ISBN: 978-0-470-27858-1.
[72] H. Terashima-Marín, P. Ross, M. Valenzuela-Rendón, Clique-based crossover for solving the timetabling problem with GAs, in: The Proceeding of the Congress on Evolutionary Computation, Washington, DC, 1999, pp. 1200–1206.
[73] J.M. Thompson, K.A. Dowsland, Variants of simulated annealing for the examination timetabling problem, Annals of Operations Research 63 (1996) 105–128.
[74] H. Turabieh, S. Abdullah, A hybrid fish swarm optimisation algorithm for solving the examination timetabling problem, in: Learning and Intelligent Optimisation Workshop (LION 5), Lecture Notes in Computer Science, vol. 6683, Springer, pp. 539–551.
[75] H. Turabieh, S. Abdullah, B. McCollum, P. McMullan, Fish swarm intelligent algorithm for the course timetabling problem, in: J. Yu et al. (Eds.), RSKT 2010, LNAI 6401, Springer-Verlag, Berlin, 2010, pp. 582–589.
[76] H. Turabieh, S. Abdullah, An integrated hybrid approach to the examination timetabling problem, OMEGA – The International Journal of Management Science 39 (6) (2011) 598–607.
[77] O. Ulker, E. Ozcan, E.E. Korkmaz, Linear linkage encoding in grouping problems: applications on graph colouring and timetabling, in: E.K. Burke, H. Rudova (Eds.), Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 3867, Springer, 2007, pp. 347–363.
[78] G.M. White, B. S Xie, Examination timetables and tabu search with longer-term memory, in: E.K. Burke, W. Erben (Eds.), Selected Papers from 3rd International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, vol. 2079, Springer, 2001, pp. 85–103.