Available online at www.sciencedirect.com
ScienceDirect
Computers & Operations Research 36 (2009) 981 – 1001
computers & operations research
www.elsevier.com/locate/cor

An investigation of fuzzy multiple heuristic orderings in the construction of university examination timetables

Hishammudin Asmuni[a], Edmund K. Burke[b], Jonathan M. Garibaldi[b,*], Barry McCollum[c], Andrew J. Parkes[b]

[a] *Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia*
[b] *Automated Scheduling, Optimisation and Planning (ASAP) Research Group, School of Computer Science, University of Nottingham, Nottingham, NG8 1BB, UK*
[c] *School of Computer Science, Queen's University Belfast, Belfast BT7 1NN, UK*

Available online 15 December 2007

Abstract

In this paper, we present an investigation into using fuzzy methodologies to guide the construction of high quality feasible examination timetabling solutions. The provision of automated solutions to the examination timetabling problem is achieved through a combination of construction and improvement. The enhancement of solutions through the use of techniques such as metaheuristics is, in some cases, dependent on the quality of the solution obtained during the construction process. With a few notable exceptions, recent research has concentrated on the improvement of solutions as opposed to focusing on investigating the 'best' approaches to the construction phase. Addressing this issue, our approach is based on combining multiple criteria in deciding on how the construction phase should proceed. Fuzzy methods were used to combine three single construction heuristics into three different pair wise combinations of heuristics in order to guide the order in which exams were selected to be inserted into the timetable solution. In order to investigate the approach, we compared the performance of the various heuristic approaches with respect to a number of important criteria (overall cost penalty, number of skipped exams, number of iterations of a *rescheduling procedure* required and computational time) on 12 well-known benchmark problems. We demonstrate that the fuzzy combination of heuristics allows high quality solutions to be constructed. On one of the 12 problems, we obtained lower penalty than any previously published constructive method and for all 12 we obtained lower penalty than when any of the single heuristics were used alone. Furthermore, we demonstrate that the fuzzy approach used less backtracking when constructing solutions than any of the single heuristics. We conclude that this novel fuzzy approach is a highly effective method for heuristically constructing solutions and, as such, has particular relevance to real-world situations in which the construction of feasible solutions is often a difficult task in its own right.

Keywords: Examination timetabling; Fuzzy methodologies; Sequential construction

1. Introduction

University timetabling is a significant administrative issue that arises in academic institutions. Such problems are well studied in the academic literature and are well known to represent difficult problems to solve [1,18–25,44]. In general, the problem is concerned with the goal of allocating a time slot for all events within a limited number of time

* Corresponding author. Tel.: +44 115 951 4216; fax: +44 115 951 4254.
E-mail address: jmg@cs.nott.ac.uk (J.M. Garibaldi).

0305-0548/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2007.12.007

slots subject to certain constraints. Basically there are two types of constraints—hard constraints and soft constraints. Hard constraints need to be satisfied under any circumstances, whereas the satisfaction of soft constraints is desirable but not absolutely necessary. Different universities emphasise different sets of constraints that reflect their needs. In the context of exam timetabling, Burke et al. reported a variety of constraints that have been implemented by universities in the United Kingdom [2].

In the literature, there has been a wide investigation of automated timetabling approaches from across the artificial intelligence and operational research communities. Approaches such as Evolutionary Algorithms [2–5], Tabu Search [6–9], Simulated Annealing [10], Constraint Programming [11–13], Case Based Reasoning [14,15] and Fuzzy Methodologies [14,16,17] have been successfully applied to timetabling problems. For more details about the variety of approaches that have been investigated, the interested reader can consult a number of survey and overview articles [1,18–24,44]. Timetabling problems can be broadly split into two categories: Course Timetabling and Exam Timetabling [43]. This paper concentrates upon exam timetabling. A detailed recent overview of the exam timetabling literature can be seen in [44].

In our previous work [16], we investigated the use of fuzzy techniques to consider two pairs of heuristics simultaneously, by using these combinations of heuristics to order exams based on an assessment of how difficult they are to schedule. This ordering was then used to construct timetable solutions through a sequential constructive algorithm. We demonstrated that certain fuzzy combinations could outperform any single heuristic ordering on the benchmark data sets used. Encouraged by the findings, we now extend this work in order to explore all three pair-wise combinations of the same three heuristics. Furthermore, we investigate the effect of combining these multiple heuristics by considering three criteria in the construction phase which reflect the effectiveness of the construction technique. These are the computational time, the number of 'skipped exams', and the number of times a *rescheduling procedure* is required. As the algorithms include a stochastic element, the experiments were repeated a number of times in order to gain a representative view of the different approaches. These experiments have provided more evidence to support our hypothesis that considering multiple heuristic orderings simultaneously can guide a sequential constructive algorithm towards better solutions. Note that techniques for the iterative improvement of the solutions that are constructed are not covered in this work. However, the method that we present does represent a quick and effective procedure for producing the initial solutions for such approaches.

The rest of this paper is organised as follows. Firstly, a formal description of the specific examination timetabling problem considered in this paper is given. Then, in Sections 3–5, the sequential constructive algorithm and fuzzy approach are explained. The computational experiments, including description of the data sets used, experimental methods and results, are given in Section 6. In Section 7, discussion and analysis of the results are presented. Finally, the conclusions are given in Section 8.

2. Problem description

Examination timetabling is essentially the problem of allocating exams to a limited number of time periods in such a way that none of the specified hard constraints are violated. A timetable which satisfies all hard constraints is termed a *feasible* timetable. In addition to the hard constraints, there are often many soft constraints whose satisfaction is desirable (but not essential). The set of constraints which need to be satisfied is usually very different from one institution to another, as reported by Burke et al. [25]. In practice, each institution usually has a different way of evaluating the quality of a feasible timetable. In many cases, the measure of quality is calculated based upon a penalty function which represents the degree to which the soft constraints are violated. Details of constraints for examination timetabling can be found in [6,25], for example.

2.1. Problem formulation

Several models and formulations for timetabling problems have been presented by various researchers. The formulation used in this research is as follows.

Indices:

$m, n = 1, \ldots, N$ examination indices;

$p, q = 1, \ldots, P$ period (time slot) indices.

Decision variables:

$$t_{np} = \begin{cases} 1 & \text{if exam } n \text{ is scheduled in period } p, \\ 0 & \text{otherwise.} \end{cases}$$

Parameters:

$N =$ number of examinations;

$P =$ number of periods (time slots) available;

$S =$ total number of students;

$$s_{nm} = \begin{cases} \text{number of students sitting both exam } n \text{ and } m, & n \neq m \\ 0, & n = m. \end{cases}$$

Objective function:
Minimise the penalty for examinations in close proximity, denoted as

$$\frac{1}{S} \sum_{n=1}^{N-1} \sum_{m=n+1}^{N} \sum_{p=1}^{P} \sum_{q=\max(1,p-5)}^{\min(p+5,P)} 2^{5-|p-q|} s_{nm} t_{np} t_{mq} \rightarrow \text{min!} \tag{1}$$

Subject to the constraints that:
Each exam must be assigned to exactly one period

$$\sum_{p=1}^{P} t_{np} = 1, \quad \forall n. \tag{2}$$

No student is able to attend more than one exam at the same time

$$t_{np} + t_{mp} \leqslant 1, \quad \forall n, m \text{ where } s_{nm} > 0, \quad \forall p. \tag{3}$$

2.2. The objective function

There are many different criteria that can be included when evaluating the quality of examination timetables, so the definition of the objective function is dependent on which criteria are to be used for the particular educational institution. The objective function specified in Eq. (1) is widely used in timetabling research to measure the timetable quality (e.g. [26]). It is often termed the 'proximity cost function', as it captures the notion of trying to spread out each student's schedule. It is often expressed (e.g. [27]) in the form

$$\frac{1}{S} \sum_{n=1}^{N-1} \sum_{m=n+1}^{N} 2^{5-|p_n - p_m|} s_{nm}, \quad \forall n, m \text{ where } |p_n - p_m| \in \{1, \ldots, 5\}$$

where p_n is the period in which exam n is scheduled (i.e. $t_{np_n} = 1$) and p_m is the period in which exam m is scheduled. In this paper, minimising this function is the objective in producing a 'good' solution; the solution with the lowest value of the objective function is considered to be the 'best' and the model producing this solution is the 'best model'. Only feasible timetables were accepted.

3. Timetable construction

In this paper, we investigate algorithms for constructing high quality solutions to the examination timetabling problem as specified above. We investigate variations of an algorithm which utilises fuzzy methodologies to combine various well-established heuristics for ordering the placement of examinations during construction. Note that finding feasible solutions is not necessarily a difficult task, given this problem formulation, but trivially constructed solutions may be of poor quality. In contrast, we aim to construct good quality solutions. Clearly, any iterative improvement technique

could subsequently be applied to the timetable solutions constructed using the fuzzy approach detailed here or any other construction technique. A goal is to move closer to mimicking the real-world situation in which the human process of timetabling is largely centred around constructing the best 'initial' timetable possible.

3.1. The basic sequential constructive approach

Sequential constructive techniques are amongst the earliest methods that have been used to tackle the examination timetabling problem in an automated way [28–30]. In this approach, the concept of 'failed first' is implemented. The basic idea is to first schedule the exams that might cause problems if they were to be scheduled later on in the process. By doing so, it would appear to be more likely that we can avoid the assignment of exams to time slots which might later lead to an infeasible solution. An infeasible solution is reached when several exams remain unscheduled because exams placed earlier have invalidated all the potentially valid time slots for the remaining exams. Essentially, the sequence of exams assigned to the timetable will affect the solution quality.

3.2. Graph-based heuristic orderings

Usually, the unscheduled exams are ordered in a sequence that represents how difficult it is thought likely to be to schedule the exams (most difficult first). One type of ordering strategy that has widely been accepted in the timetabling literature has evolved from the graph colouring problem. The timetabling problem in it simplest form (without soft constraints) can be represented as a graph colouring problem, in which the nodes represent the exams, colours represent the time slots and the edges represent the conflict between exams (see [45]). The following list describes the three graph colouring based heuristic orderings implemented in this research:

Largest degree (*LD*) *first*: Exams are ranked in descending order by the number of exams in conflict—i.e. priority is given to exams with the greatest number of exams in conflict.

Largest enrollment (*LE*) *first*: Exams are ranked in descending order by the number of students enrolled in each of the exam—i.e. exams with the highest number of students are given the highest priority.

Least saturation degree (*SD*) *first*: Exams are ranked in increasing order by the number of valid time slots remaining in the timetable for each exam—priority is given to exams with fewer time slots available.

In general, heuristic orderings are divided into two categories: *static* and *dynamic*. *Static* heuristic orderings are predetermined before the start of the assignment process and their values remain the same throughout the process. For the heuristic orderings described above, LD and LE are categorized as *static* heuristic orderings. The number of exams in conflict with each exam and the number of students enrolled for each exam only need to be calculated once by analysing the specific problem structure. On the other hand, SD is considered to be a *dynamic* heuristic ordering because the number of valid time slots available for unscheduled exams may change every time an exam is assigned to a valid time slot; in which case, the unscheduled exams need to be reordered.

3.3. Construction and improvement

Fig. 1 depicts a general framework for finding acceptable solutions to a timetabling problem, in which the construction process is termed 'Phase 1' and the improvement process 'Phase 2'. Normally, in Phase 1, an initial solution (i.e. a starting point for Phase 2) is constructed by using a sequential construction algorithm.

A sequential constructive algorithm requires the following steps to assign all exams to time slots:

Process 1. *Choose heuristic ordering*: In order to determine the sequence in which exams are scheduled to a valid time slot, we have to decide what heuristic ordering is to be employed. Usually, any of the heuristic orderings described earlier can be employed on its own to measure the exams' difficulty to be scheduled. In this paper, we describe an alternative in which two heuristic orderings are considered simultaneously to measure the exams' difficulty.

Process 2. *Calculate exams' difficulty to be scheduled*: Having chosen a heuristic ordering to be implemented, the calculations of heuristic assessment of difficulty are performed and the exams are ordered in the specified sequence.

Process 3–*Process* 5. *Sequentially assign exams to time slots*: For each exam in turn (starting with the most difficult to schedule) the following is carried out. The free time slots are examined in turn to find valid ones and, for each, the penalty that would result from placement of the exam in this slot is calculated. After examining each of the time slots, the exam is scheduled into the available slot incurring the least penalty (if two or more slots share the lowest penalty

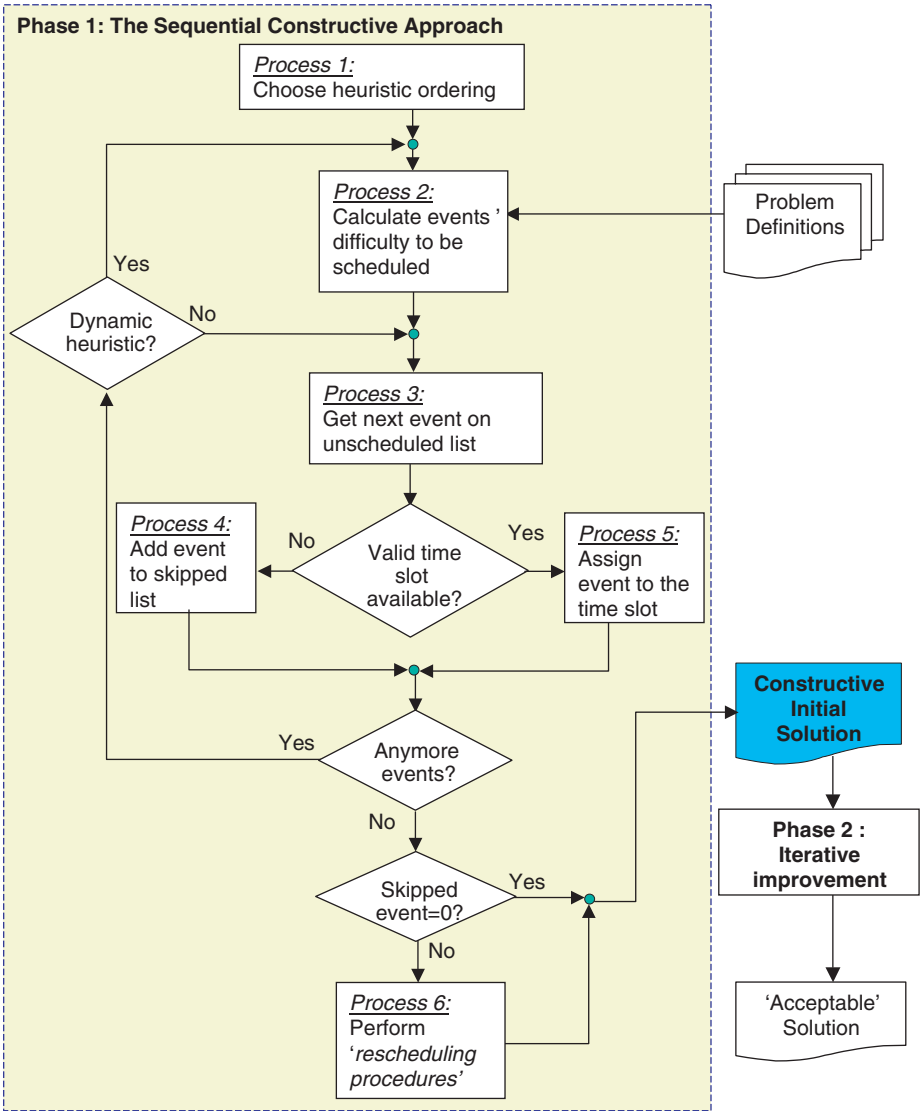

Fig. 1. A general framework for producing timetabling solutions.

cost, the exam is scheduled into the last such time slot). If no valid time slot is available, the exam is not assigned and is recorded on a 'skipped list'. If a dynamic heuristic is being used, the remaining exams' difficulties are updated and the exams are reordered accordingly.

Process 6. *Perform a* '*rescheduling procedure*': This process is only performed when there is at least one exam that could not be scheduled because no valid time slot was available—i.e there are skipped exam(s) from *Process* 3. The process for scheduling the skipped exams is shown in Fig. 2.

These processes continue until all the exams are scheduled, i.e. until a feasible solution is constructed. Although in some approaches infeasible solutions are initially accepted (usually to be later 'corrected' during an iterative improvement phase), only feasible solutions are accepted in our implementation.

In Phase 2, the initial solution is modified in order to improve the solution. The improvements can be implemented by using any meta heuristic search technique such as Genetic Algorithms, Tabu Search, Simulated Annealing [31] or the Great Deluge Algorithm [32] (to name just a few approaches).

In this paper, we focus only on constructing a feasible solution. Our main aim was to investigate the implementation of a fuzzy approach in considering multiple heuristic orderings for measuring the difficulty of scheduling exams into

```
while there exist unscheduled events
    E* = next event that needs to be scheduled;
    find time slots where event E* can be inserted which have the minimum number of
        scheduled events that need to be removed from the time slot;
    if more than one time slot is found with the same number of scheduled events to be removed
        select a time slot t randomly from the candidate list;
    end if
    while there exist events that conflict with event E* in time slot t
        Et = next conflicted event in time slot t;
        if another time slot found with same minimum penalty cost as event Et
            move event Et to that time slot;
        else
            bump back event Et to unscheduled events list;
        end if
    end while
    insert event E* to time slot t;
    remove event E* from unscheduled event list;
    if dynamic ordering is in use
        sort unscheduled events using selected heuristic ordering;
    end if
end while
```

Fig. 2. Pseudo code for the *rescheduling procedure*, used if 'skipped' exams are present.

time slots. A previous study by Carter et al. [26] has shown that using different heuristic orderings in the constructive algorithm will affect the performance of that algorithm. Their study indicated that it is not easy to determine which heuristic ordering is the most appropriate (to construct an initial solution that leads to the best iterative improvement) for any given problem in hand.

4. Combining heuristic orderings

4.1. The need to combine ordering heuristics

As stated in the previous section, when deciding the order of exams to be placed in a timetable, we are dealing with a decision making problem based on more than one attribute (or factor). The problem lies in deciding which attribute should be emphasized in order to obtain the best decision. Often it is difficult to resolve conflicting attributes. Consider the example shown in Fig. 3. In this example, there are 10 exams ($e1$, $e2$, $e3$, $e4$, $e5$, $e6$, $e7$, $e8$, $e9$, $e10$) with the given LD and LE values. Fig. 3(a) shows the 10 exams in an unordered list, Figs. 3(b)–(f) show the results of using different heuristic orderings to order the 10 exams. It can be seen that when two different heuristic orderings are used individually, the orderings are different (see Figs. 3(b) and (c)).

The simplest method to handle such multiple attribute decision making is just to multiply the value of each attribute by a weighting factor and summate (i.e. form a simple linear combination) In this example, the formulation is just:

$$weight(e_j) = w_l LD_j + w_e LE_j,$$

where $j = 1, 2, \ldots, n$; n is the number of exams; and w_l and w_e are the weighting factors (any real number) for LD and LE, respectively. Using a simple combination to represent the relative importance of both attributes can result in quite a different ordering (see Fig. 3(e) where weights $w_l = 0.45$ and $w_e = 0.55$ have been used as an arbitrary example). In effect, neither the LD nor LE attributes alone control the exam ordering; it is determined by considering both attributes simultaneously. However, the problem then becomes that of needing to search for the appropriate values of w_l and w_e to be used. Johnson implemented the formula above [33] for constructing initial solutions although he actually set the w_l weight to a constant value ($w_l = 1$) while varying the w_e value. The aim of this was simply to produce a range of alternative initial solutions which were then subjected to iterative improvement.

4.2. The fuzzy approach

Heuristic orderings are based on assumptions such as, for example, an exam is more difficult to schedule if it has a 'large' number of other exams in conflict or if it has a 'small' number of valid time slots available. This is dealing with linguistic terms, where no exact values for 'large' and 'small' have been defined. The general framework of

a Unordered exams list

exams	*LD*	*LE*
e1	30	40
e2	10	30
e3	50	20
e4	20	35
e5	39	10
e6	10	43
e7	10	20
e8	19	25
e9	27	15
e10	45	30

b Ordered by *LD* only

exams	*LD*	*LE*
e3	50	20
e10	45	30
e5	39	10
e1	30	40
e9	27	15
e4	20	35
e8	19	25
e2	10	30
e6	10	43
e7	10	20

c Ordered by *LE* only

exams	*LD*	*LE*
e6	10	43
e1	30	40
e4	20	35
e2	10	30
e10	45	30
e8	19	25
e7	10	20
e3	50	20
e9	27	15
e5	39	10

d Ordered by *LD* and then *LE*

exams	*LD*	*LE*
e3	50	20
e10	45	30
e5	39	10
e1	30	40
e9	27	15
e4	20	35
e8	19	25
e6	10	43
e2	10	30
e7	10	20

e Ordered by *LE* and then *LD*

exams	*LD*	*LE*
e6	10	43
e1	30	40
e4	20	35
e10	45	30
e2	10	30
e8	19	25
e3	50	20
e7	10	20
e9	27	15
e5	39	10

f Ordered by linear combination of both attributes

exams	*LD*	*LE*	weight
e10	45	30	36.8
e1	30	40	35.5
e3	50	20	33.5
e4	20	35	28.3
e6	10	43	28.2
e5	39	10	23.1
e8	19	25	22.3
e2	10	30	21.0
e9	27	15	20.4
e7	10	20	15.5

Fig. 3. Example of examinations ordered by various combinations of heuristics.

fuzzy reasoning facilitates the handling of such uncertainty. Since being first introduced by Zadeh in 1965 [34], fuzzy logic approaches have been widely applied in variations of real world problem domains. Fuzzy systems are used for representing and employing knowledge that is imprecise, uncertain, or unreliable. Thus, our original hypothesis was that this problem might be one where fuzzy techniques fit well.

Fig. 4 shows the interconnected components of a fuzzy inference system. The fuzzification component computes the membership grade for each crisp input variable based on the membership functions defined. The rule base component consists of a set of rules that connect input variables to output variables in 'IF …THEN …' form. These are used to describe the desired system response in terms of *linguistic* variables (words) rather than mathematical formulae. The 'IF' part of the rule is referred to as the 'antecedent', the 'THEN' part is referred to as the 'consequent'. The number of rules depends on the number of inputs and outputs, and the desired behaviour of the system. The inference engine then conducts the fuzzy reasoning process by applying the appropriate fuzzy operators in order to obtain the fuzzy set to be accumulated in the output variable. The defuzzifier transforms the output fuzzy set to crisp output by applying the specific defuzzification method.

More formally, a fuzzy set A of a universe of discourse X (the range over which the variable spans) is characterised by a *membership function* $\mu_A : X \to [0, 1]$ which associates with each element x of X a number $\mu_A(x)$ in the interval

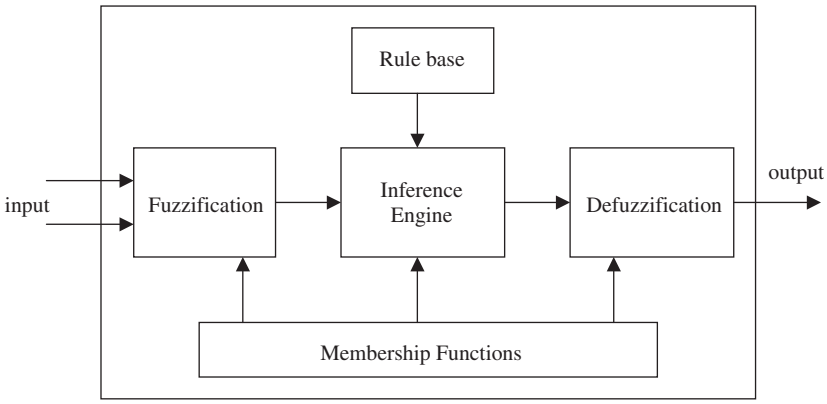

Fig. 4. The components of a fuzzy inference system.

[0, 1], with $\mu_A(x)$ representing the *grade of membership* of x in A. The precise meaning of the membership grade is not rigidly defined, but is supposed to capture the 'compatibility' of an element to the notion of the set. There are many alternatives for implementing the general fuzzy reasoning methodology. In our implementation, the common Mamdani style fuzzy inference was used with standard Zadeh (min-max) operators. In Mamdani inference [35], rules are of the following form:

$$R_i : \text{if } (x_1 \text{ is } A_{i,1}) \text{ and } (x_2 \text{ is } A_{i,2}) \ldots \text{ and } (x_r \text{ is } A_{i,r}) \text{ then } (y \text{ is } C_i) \text{ for } i = 1, 2, \ldots, L,$$

where L is the number of rules, x_j $(j = 1, 2, 3, \ldots, r)$ are input variables, y is the output variable, and A_{ij} and C_i are fuzzy sets that are characterised by membership functions $A_{ij}(x_j)$ and $C_i(y)$, respectively. The final output of a Mamdani system is one or more arbitrarily complex fuzzy sets which (usually) need to be defuzzified. We applied a common form of this process, termed 'centre of gravity defuzzification', as it is based upon the notion of finding the centroid of a planar figure, as given by

$$\sum_i \frac{\mu(x_i) \cdot x_i}{\mu(x_i)}.$$

It is not appropriate to present a full description of the functioning of fuzzy systems here; the interested reader is referred to [35] for a simple treatment or [36] for a more complete treatment. An introductory tutorial on fuzzy modelling for the novice can be found in [31] and [37].

5. A fuzzy model for timetable construction

This section presents the development of our particular fuzzy model. Considering the three single heuristic orderings explained in Section 3.2, there are three alternatives in which two single heuristic orderings can be simultaneously combined. The possible combinations are:

- LD and LE, referred to as the *Fuzzy LD+LE Model* in the rest of this paper.
- Saturation Degree (SD) and Largest Enrollment (LE), referred to as the *Fuzzy SD+LE Model* in the rest of this paper.
- Saturation Degree (SD) and Largest Degree (LD), referred to as the *Fuzzy SD+LD Model* in the rest of this paper.

As mentioned earlier, these three heuristic ordering combinations provide alternative ways for ordering the exams. Therefore, in *Process* 1 (see Fig. 1), instead of simply choosing any one of the single heuristic orderings to be implemented, we need to modify/improve the process so that the fuzzy approach can be incorporated. Accordingly, the extended version of *Process* 1 is shown in Fig. 5. It is worth mentioning that fuzzy methodologies are *only* employed in *Process* 1; the other processes in the dotted-box of Fig. 1 remain the same.

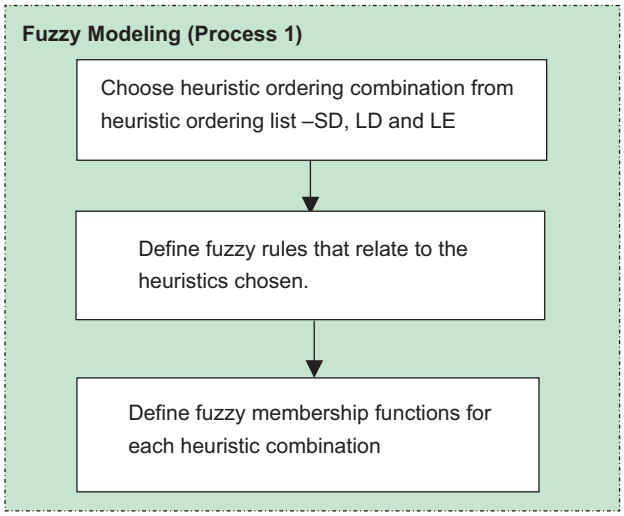

Fig. 5. The steps involved in a fuzzy version of *Process* 1 (from Fig. 1).

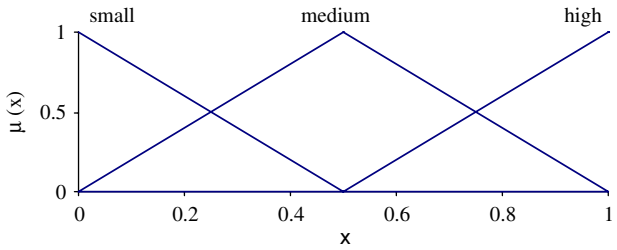

Fig. 6. Illustration of the default membership functions used in each linguistic variable.

Fuzzy modeling can be thought of as the task of designing the fuzzy inference system specific to the particular application area. The selection of important parameters for the inference system is crucial, as the overall system behaviour is highly dependent on a large number of factors such as how the membership functions are chosen, the number of rules involved, the fuzzy operators used, and so on. As we are combining two heuristics into a single overall heuristic, a fuzzy system with two inputs and one output is developed. The input variables used are dependent on the heuristic combinations selected. Three pairs of input variables are possible, namely LD and LE, SD and LE, or SD and LD. In any pair of input variables, an output variable called *examweight* is generated. This output variable, *examweight* represents the overall difficulty of scheduling an exam to a time slot. Each of the input and output variables are associated with three linguistic terms: *small*, *medium* and *high*. Each linguistic term is represented by a fuzzy membership function. Although, in general, any shape of fuzzy membership function is possible, triangular ones are popular due to their relative simplicity. We used triangular membership functions for this reason—a thorough investigation of alternative membership functions would be a major undertaking in its own right and is beyond the scope of this paper. The basic triangular membership functions implemented are shown in Fig. 6.

The range of each input and output variable was defined to be between 0 and 1. This means that the actual input value needed to be transformed into a new value in the range [0, 1]. In general, this can be achieved using a simple linear transformation:

$$v' = \frac{(v - min_x)}{(max_x - min_x)},$$

where v is the actual value in the initial range $[min_x, max_x]$. In our case, min_x was set to zero for each of LD, LE and SD. The maximum values were set by inspection of each problem instance: $max_x(LD)$ was set to the largest number of

Table 1
The fuzzy rule set for the *Fuzzy LD+LE Model*

		LE		
		S	M	H
LD	S	VS	S	M
	M	S	M	H
	H	M	H	VH

VS: very small; S: small; M: medium; H: high; VH: very high.

Table 2
The fuzzy rule set for the *Fuzzy SD+LE Model*

		SD		
		S	M	H
LE	S	M	S	VS
	M	H	M	S
	H	VH	H	M

VS: very small; S: small; M: medium; H: high; VH: very high.

Table 3
The fuzzy rule set for the *Fuzzy SD+LD Model*

		SD		
		S	M	H
LD	S	M	S	VS
	M	H	M	S
	H	VH	H	M

VS: very small; S: small; M: medium; H: high; VH: very high.

conflicts found for any exam in the problem instance; $max_x(LE)$ was set to the maximum number of students enrolled to any exam in the problem instance; and $max_x(SD)$ was set to the total number of time slots available in the problem instance.

For each heuristic ordering combination, a fuzzy rule set connecting the input variables (any two of LD, LE or SD) to the output variable, *examweight* was constructed. All three fuzzy rule sets were motivated by the assumption that exams should be placed into a timetable in order of how difficult they are to schedule (most difficult first) in accordance with the respective heuristics LD, LE and SD. These assumptions were used in order to get a symmetric, balanced set of fuzzy rules for each heuristic ordering combination, to ensure that all possible input values were covered. Note that the interpretation of the SD heuristic (smaller is more difficult) is linguistically opposite to that of LD and LE (larger is more difficult). Thus, care must be taken when considering SD as one of the heuristic orderings in a combination.

The fuzzy rules sets for the *Fuzzy LD+LE Model*, *Fuzzy SD+LE Model* and *Fuzzy SD+LD Model* are shown in Tables 1–3, respectively. For simplicity, the fuzzy rules are expressed as a linguistic matrix (see [38]). In such a linguistic

matrix, the left-most column and the first row denote the variables involved in the antecedent part of the rules. The second column contains the linguistic terms applicable to the input variable shown in the first column; those in the second row correspond to the input variable shown in the first row. Each entry in the main body of the matrix denotes the linguistic values of the consequent part of a rule.

Note that, in addition to the three basic terms, the *hedge* 'very' was utilised to create two extra terms for the output variable. The 'very' hedge squares the membership grade $\mu(x)$ at each x of the fuzzy set for the term to which it is applied. Thus the membership function of the fuzzy set for 'very small' is obtained by squaring the membership function of the fuzzy set 'small'. For instance, the bottom-right entry in Table 1 is read as "*IF LD is high AND LE is high THEN examweight is very high*". For an illustrative example of how these initial fuzzy models were tuned to best measure the difficulty of scheduling exams to time slots by considering two heuristic orderings simultaneously, the interested reader is referred to our previous paper [16].

6. Computational experiments

6.1. Methods

Due to the randomness in the *rescheduling procedure*, a different timetable may be constructed each time the algorithm is run. Therefore, in order to determine and compare the performance of the various fuzzy heuristic orderings, repeated runs were performed to generate 30 solutions with each fuzzy multiple heuristic ordering model and each of the single heuristic orderings (LD, LE and SD), for each of the 12 data sets mentioned in Section 2.

For fuzzy multiple heuristic orderings, the 'best' fuzzy models that had been identified during the membership functions tuning phase were utilised. In the tuning phase, the membership functions were refined by adjusting them until the best possible system performance was achieved. In brief, a parameter *cp* was used to represent (simultaneously) the right-hand edge of the 'small' term, the middle of the 'medium' term and the left-hand edge of the 'large' term for each of the two inputs and the output variable (see Fig. 7). The three *cp* parameters were systematically altered while assessing the performance of the system. For a detailed description of the fuzzy membership function tuning process, please refer to our previous paper [16]. Table 4shows the values of the *cp* parameter obtained for each of the fuzzy heuristic ordering combinations for each data set. Graphical representations of the membership functions for some of the 'best' fuzzy models generated when the *Fuzzy SD+LE Model* was implemented are depicted in Fig. 8.

6.2. Problem instances

The examination timetabling problem data sets which were made publicly available by Carter et al. [26], were used in these experiments. The 12 instances that we address have different characteristics and various levels of complexity, which are shown in Table 5. We employ the notation introduced in [44]. For all data sets, it is required to satisfy the defined hard constraint that no student can attend more than one exam at the same time. In addition, the solution must be developed in such a way that it promotes the spreading out of each student's exams so that students have as much time as possible between exams.

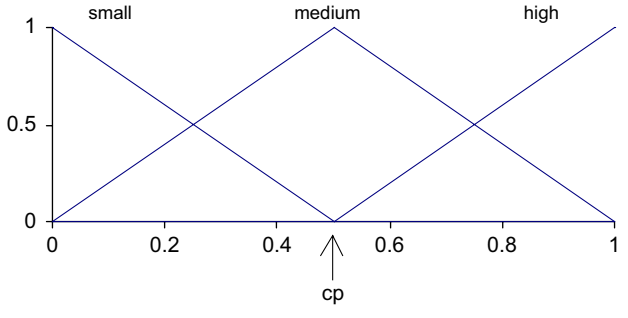

Fig. 7. The fuzzy membership functions were tuned by adjusting the *cp* parameter.

Table 4
Values for *cp* parameters obtained from the fuzzy tuning process

Data set	Fuzzy *LD+LE* Model			Fuzzy *SD+LE* Model			Fuzzy *SD+LD* Model		
	LD	LE	Examweight	SD	LE	Examweight	SD	LD	Examweight
CAR-F-92-I	0.00	0.50	0.50	0.50	0.25	0.25	0.50	1.00	1.00
CAR-S-91-I	0.50	0.00	0.25	0.25	0.00	0.50	0.75	0.75	0.75
EAR-F-83-I	0.40	1.00	0.30	0.50	1.00	0.50	0.80	0.40	0.20
HEC-S-92-I	0.40	0.20	0.40	0.40	1.00	1.00	0.20	0.40	0.00
KFU-S-93	0.75	0.00	0.00	0.50	1.00	0.50	0.50	1.00	0.50
LSE-F-91	0.75	0.50	0.25	0.25	1.00	0.25	0.25	0.75	0.50
RYE-F-92	0.75	0.25	0.00	1.00	0.00	0.00	0.75	1.00	0.50
STA-F-83-I	0.60	0.70	0.90	0.20	0.30	0.00	0.60	0.80	0.50
TRE-S-92	0.00	0.50	0.40	0.60	1.00	0.20	0.20	0.30	0.10
UTA-S-92-I	0.00	0.50	0.75	0.25	0.00	0.50	0.50	0.50	0.75
UTE-S-92	0.30	0.60	0.00	0.30	0.90	0.70	0.40	0.00	0.50
YOR-F-83-I	0.90	1.00	0.00	0.60	0.80	0.70	0.00	0.00	0.50

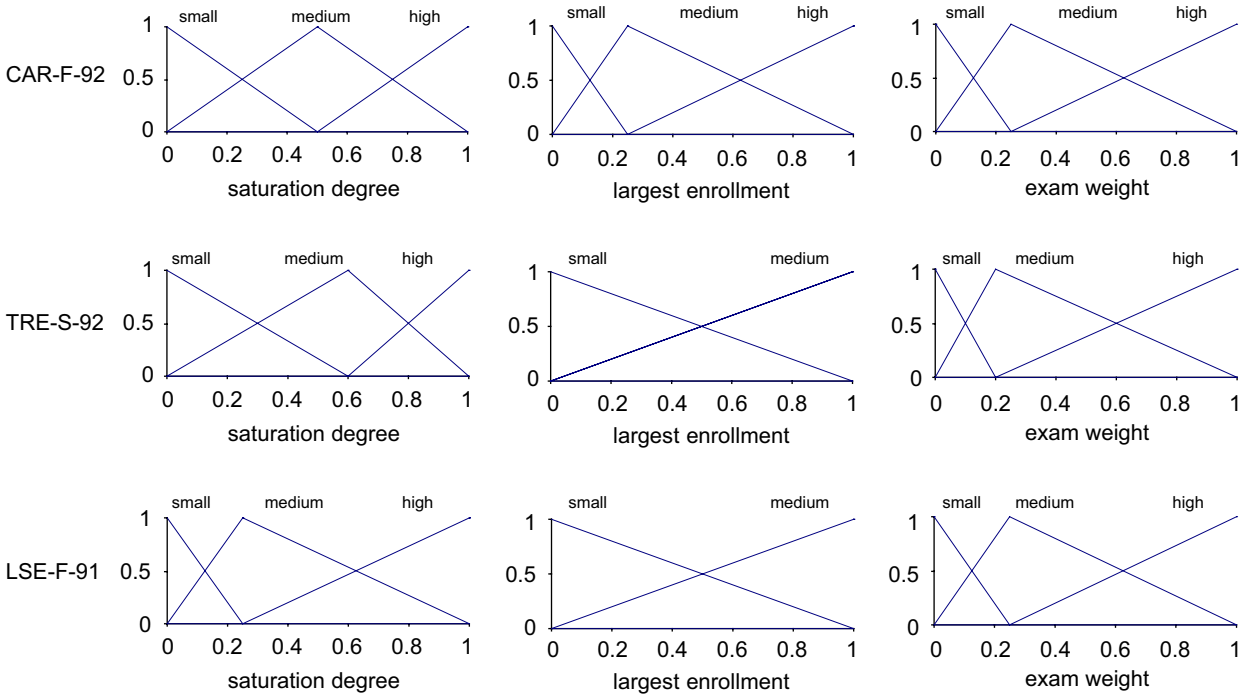

Fig. 8. Example of graphical representation of fuzzy membership functions when implemented in the *Fuzzy SD+LE Model*.

6.3. Experimental results

Table 6 shows a comparison of the cost penalties obtained based on 30 runs of each data set. The best results among the different heuristic orderings used are highlighted in bold font. It is evident that, overall, the fuzzy multiple heuristic orderings have outperformed any of the single heuristic orderings in that, for each data set, a fuzzy ordering obtained the best constructed timetable quality. Specifically, the *Fuzzy SD+LE Model* obtained 10 best results and the *Fuzzy LD+LE Model* and *Fuzzy SD+LD Model* each obtained one best result. Amongst the single heuristic orderings, LE is

Table 5
Examination timetabling problem characteristics

Data set	Number of slots	Number of exams	Number of students	Conflict density
CAR-F-92-I	32	543	18 419	0.14
CAR-S-91-I	35	682	16 925	0.13
EAR-F-83-I	24	190	1125	0.27
HEC-S-92-I	18	81	2823	0.42
KFU-S-93	20	461	5349	0.06
LSE-F-91	18	381	2726	0.06
RYE-F-92	23	486	11 483	0.08
STA-F-83-I	13	139	611	0.14
TRE-S-92	23	261	4360	0.18
UTA-S-92-I	35	622	21 266	0.13
UTE-S-92	10	184	2750	0.08
YOR-F-83-I	21	181	941	0.29

the most effective on these benchmarks in that it obtained eight best results, followed by SD with three best results (*HEC-S-92-I*, *LSE-F-91* and *UTA-S-92-I*) and lastly LD with only one best result (*EAR-F-83-I*).

Table 7 shows a comparison of the best result obtained for each data set by the fuzzy approach detailed here with the best results in the literature for other *purely constructive* methods (marked with a †) and the best current iterative improvement approaches. We define a purely constructive approach as one which stops as soon as a feasible solution has been created. However, recall that we aim to construct a *good quality* solution, rather than just constructing *any* solution. It can be seen that for one data set, *YOR-F-83-I*, the fuzzy approach obtains the best overall result for any constructive approach. Whilst it can also be seen that our methodology does not produce any best overall results, we reiterate that any iterative improvement technique could subsequently be applied to the timetable solutions constructed using the fuzzy approach detailed here.

Table 8 shows the number of skipped exams obtained before the *rescheduling procedure* was called. Recall that the number of skipped exams is the number of exams that could not be scheduled after the completion of the initial phase of the constructions process (i.e. after *Process* 2–*Process* 5 had been completed). It is simply the number of exams added to the 'skipped list' due to the fact that no valid time slot was available. It can be seen that SD often (seven out of twelve data sets) produced solutions without any skipped exams. This behaviour (most data sets resulting in no skipped exams) is also seen in the fuzzy multiple heuristic orderings that used SD. However, this was not true for two data sets (*RYE-F-92* and *STA-F-83-I*) when the *Fuzzy SD+LD Model* was implemented—i.e. for these two data sets the SD heuristic alone resulted in no skipped exams, but when combined with the LD heuristic in the fuzzy approach some exams were skipped. The number of skipped exams determines whether it is necessary to invoke the *rescheduling procedure* or not. Obviously, it is not necessary to invoke the *rescheduling procedure* if there are no skipped exams.

Table 9 shows a comparison of the number of iterations of the *rescheduling procedure* required. This table shows the number of iterations of the loop in the *rescheduling procedure* that were required by each heuristic ordering in order to produce the solutions. As mentioned earlier, the number of skipped exams has an effect on the number of iterations of the *rescheduling procedure* that are required. In our approach, the *rescheduling procedure* is invoked at least once for every exam in the unscheduled list, so that the number of iterations of the *rescheduling procedure* required is equal to or greater than the number of skipped exams (although this need not *necessarily* be the case in general). For example, when LD ordering was applied to the *YOR-F-83-I* data set, it caused five skipped exams (see column 2 of Table 8). However, on average, 27 iterations of the *rescheduling procedure* were required (see column 2 of Table 9) in order to produce the solutions.

Finally, Table 10 shows a comparison of the computational time required to construct the solutions for each of the heuristic ordering methods for each data set. As might be expected, when dynamic heuristic ordering was used, much longer times were required in order to produce the solutions because, each time around the loop, the heuristic needed to be recalculated and the exams reordered. This happened when either a single or a multiple heuristic ordering was implemented.

Table 6
The penalty costs obtained by the different heuristic orderings on each of the 12 benchmark data sets

Data set		Single heuristic ordering			Fuzzy *LD+LE* Model	Fuzzy *SD+LE* Model	Fuzzy *SD+LD* Model
		LD	LE	SD			
CAR-F-92-I	Best	5.51	4.86	5.50	4.62	**4.54**	4.62
	Average	6.10	5.42	5.74	4.63	4.54	4.62
	Worst	6.81	6.40	7.25	4.64	4.54	4.62
	Std. dev.	0.41	0.38	0.43	0.01	0.00	0.00
CAR-S-91-I	Best	6.13	5.89	5.91	5.57	**5.29**	5.77
	Average	6.66	6.36	5.91	5.67	5.29	5.77
	Worst	7.40	6.89	5.91	5.88	5.29	5.77
	Std. dev.	0.31	0.26	0.00	0.08	0.00	0.00
EAR-F-83-I	Best	40.58	44.86	48.99	42.61	**37.02**	40.85
	Average	42.05	51.06	51.49	45.16	37.02	42.16
	Worst	45.09	59.14	54.79	49.90	37.02	44.46
	Std. dev.	1.03	2.99	1.67	1.52	0.00	1.27
HEC-S-92-I	Best	14.73	14.41	14.23	12.43	**11.78**	12.55
	Average	16.25	16.98	16.36	14.25	11.78	12.55
	Worst	18.70	21.40	20.80	18.18	11.78	12.55
	Std. dev.	1.31	1.76	1.86	1.74	0.00	0.00
KFU-S-93	Best	18.38	16.46	18.62	16.45	15.81	**15.80**
	Average	19.53	16.47	18.62	17.84	15.81	15.80
	Worst	21.81	16.50	18.62	21.75	15.81	15.80
	Std. dev.	0.94	0.01	0.00	1.64	0.00	0.00
LSE-F-91	Best	14.79	14.41	13.46	12.35	**12.09**	12.95
	Average	17.12	16.45	13.46	12.35	12.09	12.95
	Worst	19.70	18.79	13.46	12.35	12.09	12.95
	Std. dev.	1.37	1.20	0.00	0.00	0.00	0.00
RYE-F-92	Best	13.02	11.22	11.60	11.75	**10.38**	12.71
	Average	14.54	12.86	11.60	12.47	10.38	13.92
	Worst	17.38	14.60	11.60	13.70	10.38	15.42
	Std. dev.	1.10	0.84	0.00	0.52	0.00	0.69
STA-F-83-I	Best	173.09	171.80	178.24	**160.42**	160.75	171.42
	Average	173.09	172.22	178.24	160.42	160.75	171.42
	Worst	173.09	172.57	178.24	160.42	160.75	171.42
	Std. dev.	0.00	0.23	0.00	0.00	0.00	0.00
TRE-S-92	Best	10.65	9.92	10.81	9.05	**8.67**	9.80
	Average	11.42	10.73	10.81	9.05	8.67	9.80
	Worst	12.32	12.02	10.81	9.05	8.67	9.80
	Std. dev.	0.43	0.49	0.00	0.00	0.00	0.00
UTA-S-92-I	Best	4.26	4.63	3.83	3.86	**3.57**	3.86
	Average	5.14	5.31	3.83	4.03	3.57	3.86
	Worst	6.28	6.32	3.83	4.30	3.57	3.86
	Std. dev.	0.49	0.33	0.00	0.13	0.00	0.00
UTE-S-92	Best	35.19	28.79	33.26	28.65	**28.07**	31.05
	Average	35.51	28.93	33.61	28.68	28.07	31.05
	Worst	36.10	29.63	34.43	28.74	28.07	31.05
	Std. dev.	0.26	0.20	0.28	0.03	0.00	0.00
YOR-F-83-I	Best	45.32	43.33	45.26	41.02	**39.80**	44.70
	Average	46.27	45.75	46.57	43.05	39.80	44.70
	Worst	47.91	49.12	48.53	47.95	39.80	44.70
	Std. dev.	0.79	1.81	1.01	1.40	0.00	0.00

In each case the best result, the worst result, the average result and the standard deviation obtained over 30 repeated runs are given.

Table 7
A comparison of results obtained using different approaches

Data set	† Carter et al., 1996 [26]	† Burke et al., 2004 [39]	† Burke et al., 2007 [40]	† Asmuni et al., 2005 [16]	† Fuzzy Multiple Heuristic	Caramia et al., 2001 [41]	Yang and Petrovic, 2005 [14]	Burke et al., 2006 [42]
CAR-F-92-I	6.2	**4.32**	4.53	4.56	*4.54*	6.6	4.50	4.6
CAR-S-91-I	7.1	4.97	5.36	5.29	5.29	6.0	**3.93**	4.0
EAR-F-83-I	36.4	36.16	37.92	37.02	37.02	**29.3**	33.70	32.8
HEC-S-92-I	10.8	11.61	12.25	11.78	11.78	**9.2**	10.83	10.0
KFU-S-93	14.0	15.02	15.20	15.81	*15.80*	13.8	13.82	**13.0**
LSE-F-91	10.5	10.96	11.33	12.09	12.09	**9.6**	10.35	10.0
RYE-F-92	7.3	–	–	10.38	10.38	**6.8**	8.53	–
STA-F-83-I	161.5	170.35	**158.19**	160.42	160.42	158.2	158.35	159.9
TRE-S-92	9.6	8.38	8.92	8.67	8.67	9.4	7.92	**7.9**
UTA-S-92-I	3.5	3.36	3.88	3.57	3.57	3.5	**3.14**	3.2
UTE-S-92	25.8	27.42	28.01	28.07	28.07	**24.4**	25.39	24.8
YOR-F-83-I	41.7	40.77	41.37	40.66	*39.80*	**36.2**	36.35	37.3

Constructive approaches are marked with a dagger (†). The last three columns show the best current iterative improvement techniques. Bold font indicates the best result achieved by any method; underlined text indicates the best result achieved by a constructive method; italic font indicates results of our method that beat our previously published work [16].

Table 8
The number of skipped exams obtained due to the fact that there was no valid time slot available in the first attempt to assign the exam into the time slots—i.e. the number of exams in the skipped list after *Process* 5

Data set	Single heuristic ordering			Fuzzy *LD+LE* Model	Fuzzy *SD+LE* Model	Fuzzy *SD+LD* Model
	LD	LE	SD			
CAR-F-92-I	12	11	1	1	**0**	**0**
CAR-S-91-I	10	15	**0**	3	**0**	**0**
EAR-F-83-I	3	8	1	7	**0**	1
HEC-S-92-I	2	6	2	5	**1**	**0**
KFU-S-93	4	4	**0**	8	**0**	**0**
LSE-F-91	3	5	**0**	**0**	**0**	**0**
RYE-F-92	2	5	**0**	1	**0**	2
STA-F-83-I	24	2	**0**	7	**0**	24
TRE-S-92	6	7	**0**	1	**0**	**0**
UTA-S-92-I	7	13	**0**	2	**0**	**0**
UTE-S-92	2	3	**1**	**1**	**1**	**1**
YOR-F-83-I	5	10	3	13	**0**	**0**

7. Discussion

When constructing solutions for examination timetabling problems, one of the most important aspects that will affect the solution quality is the sequence in which the events should be selected to be scheduled [12]. Many ordering strategies have been implemented by other researchers. One of the strategies that is widely used is to base various heuristics on graph theory [45]. However, to the best of our knowledge, although there are many such criteria derived from graph theory that could be used as ordering heuristics, only one criterion has been used on its own at any single step of the construction process (except the work of Johnson [33] where the LE and LD heuristic were employed simultaneously simply to construct a number of alternative initial solutions). Another similar approach is one recently published by Burke et al. [40] in which different graph colouring heuristics are applied in sequence to construct solutions for both the examination and course timetabling problem, but only one heuristic is used at any given time.

This paper presents a new heuristic ordering method in which two heuristic orderings are considered simultaneously using a fuzzy methodology to combine them. The experimental results shown in Table 6 indicate that this new approach

Table 9
The number of iterations of the *rescheduling procedure* required for each data set

Data set		Single heuristic ordering			Fuzzy *LD+LE* Model	Fuzzy *SD+LE* Model	Fuzzy *SD+LD* Model
		LD	LE	SD			
CAR-F-92-I	Smallest	58	31	5	1	0	0
	Average	204	81	58	1	0	0
	Worst	459	223	261	1	0	0
CAR-S-91-I	Smallest	39	34	0	4	0	0
	Average	99	70	0	10	0	0
	Worst	287	152	0	33	0	0
EAR-F-83-I	Smallest	4	17	7	11	0	2
	Average	7	95	49	24	0	12
	Worst	12	265	167	57	0	53
HEC-S-92-I	Smallest	8	19	9	6	1	0
	Average	29	41	39	22	1	0
	Worst	101	80	121	115	1	0
KFU-S-93	Smallest	6	4	0	10	0	0
	Average	13	4	0	29	0	0
	Worst	80	4	0	117	0	0
LSE-F-91	Smallest	13	24	0	0	0	0
	Average	59	71	0	0	0	0
	Worst	182	181	0	0	0	0
RYE-F-92	Smallest	9	9		6	0	6
	Average	88	28	0	22	0	59
	Worst	365	86	0	73	0	217
STA-F-83-I	Smallest	24	2	0	7	0	24
	Average	24	2	0	7	0	24
	Worst	24	2	0	7	0	24
TRE-S-92	Smallest	12	13	0	1	0	0
	Average	38	31	0	1	0	0
	Worst	121	67	0	1	0	0
UTA-S-92-I	Smallest	37	65	0	4	0	0
	Average	186	239	0	34	0	0
	Worst	413	543	0	82	0	0
UTE-S-92	Smallest	3	3	3	1	1	1
	Average	9	3	9	1	1	1
	Worst	66	11	32	1	1	1
YOR-F-83-I	Smallest	8	18	11	14	0	0
	Average	27	60	50	33	0	0
	Worst	65	181	142	107	0	0

is promising. Concentrating on the quality of the solutions, it can be seen in Table 6 that all the best results were obtained when fuzzy multiple heuristic orderings were implemented. This indicates that, in these timetabling problems, determining the difficulty of scheduling exams into time slots by taking into account multiple heuristic orderings simultaneously has resulted in solutions with better quality. The results in Table 7 show that, for the data set *YOR-F-83-I*, the fuzzy approach obtains the best overall result for any constructive approach.

Nevertheless, there are a few cases in which fuzzy multiple heuristic orderings produced worst solutions compared to specific single heuristic orderings. For example, for the *RYE-F-92*, *UTE-S-92* and *YOR-F-83-I* data sets the LE heuristic ordering beat the *Fuzzy SD+LD Model* (see Table 6), and there are other similar such occurrences. These observations

Table 10
A comparison of the computational time required to construct the solutions for each heuristic ordering methods for each data set (seconds)

Data set		Single heuristic ordering			Fuzzy *LD+LE* Model	Fuzzy *SD+LE* Model	Fuzzy *SD+LD* Model
		LD	LE	SD			
CAR-F-92-I	Shortest	45.09	20.50	396.30	2.13	442.98	725.08
	Average	185.27	70.50	446.86	2.18	446.77	733.13
	Worst	440.08	216.67	666.81	2.67	458.31	763.75
CAR-S-91-I	Shortest	47.16	36.08	922.58	6.06	1006.70	1620.36
	Average	135.72	87.14	965.61	14.16	1023.50	1653.55
	Worst	403.24	197.70	1161.44	49.66	1055.53	1767.08
EAR-F-83-I	Shortest	0.41	1.13	12.61	0.83	19.34	33.34
	Average	0.63	8.26	16.62	1.88	19.38	34.82
	Worst	1.13	23.74	27.70	4.88	19.47	40.77
HEC-S-92-I	Shortest	0.11	0.22	0.95	0.11	2.28	2.27
	Average	0.37	0.52	1.29	0.32	2.36	2.36
	Worst	1.33	1.03	2.36	1.45	3.49	3.49
KFU-S-93	Shortest	1.17	0.89	64.28	2.05	112.44	179.50
	Average	2.74	0.91	64.54	7.77	113.92	182.91
	Worst	17.19	0.97	67.03	31.64	115.30	187.50
LSE-F-91	Shortest	1.77	3.24	37.92	0.52	70.27	114.55
	Average	8.25	9.77	38.00	0.53	70.57	118.33
	Worst	27.50	24.33	38.61	0.58	70.88	136.47
RYE-F-92	Shortest	2.94	2.84	149.94	2.11	215.24	333.50
	Average	22.68	7.54	150.44	6.01	221.01	359.11
	Worst	96.94	22.64	151.75	19.55	246.77	417.64
STA-F-83-I	Shortest	0.19	0.05	3.33	0.16	6.58	7.66
	Average	0.21	0.06	3.34	0.16	6.59	9.72
	Worst	0.27	0.14	3.39	0.22	6.64	11.05
TRE-S-92	Shortest	1.08	1.31	30.02	0.47	43.55	75.39
	Average	4.12	3.57	30.08	0.49	43.70	76.94
	Worst	12.77	8.34	30.23	0.55	44.86	85.88
UTA-S-92-I	Shortest	39.38	71.22	597.94	4.95	675.06	1101.94
	Average	229.01	296.84	639.26	40.40	695.52	1111.75
	Worst	501.64	697.88	809.13	93.91	818.70	1160.22
UTE-S-92	Shortest	0.06	0.08	4.23	0.14	12.67	18.41
	Average	0.11	0.09	4.32	0.17	13.02	19.51
	Worst	0.41	0.23	4.95	0.39	13.33	24.52
YOR-F-83-I	Shortest	0.42	0.88	15.99	0.78	22.47	37.22
	Average	1.34	3.06	18.03	1.74	22.51	38.78
	Worst	3.17	9.39	23.53	5.16	22.59	46.23

suggest that care must be taken when choosing which heuristic orderings are to be used simultaneously for any given problem instance.

When looking at 'effectiveness' in terms of both solution quality and variation in solution quality, the results indicate that the *Fuzzy SD+LE Model* is the most effective heuristic ordering. For all 12 data sets, the 30 multiple runs of this heuristic ordering obtain the same solution quality. Although the *Fuzzy SD+LD Model* also managed to obtain the same solution quality for 10 data sets, this fuzzy model only produced one best result out of the 12 data sets. Meanwhile, SD ordering and the *Fuzzy LD+LE Model* only managed to produce the same solution for a few of the data sets, while LD ordering only managed to obtain the same solution quality for the *STA-F-83-I* data set.

Since the only stochastic element in our algorithm is when selecting a time slot in the *rescheduling procedure*, any heuristic ordering that produces an exam ordering which causes no skipped exams will always obtain the same solution in multiple runs. On the other hand, in situations where there are skipped exams (which depends on the problem instance and the heuristic ordering used) these can only be scheduled by reshuffling the already scheduled exams into another time slot, or 'bumping' the scheduled exams back to the unscheduled exam list. It seems obvious that the higher the number of iterations of the *rescheduling procedure* required, the higher the possibility of obtaining a solution with a *different* cost penalty. This scenario may explain why the fuzzy membership function tuning process took a long time to finish, particularly for the problem instances that have more than 400 exams. It is assumed that during the fuzzy model tuning process, when a bad fuzzy model is evaluated, it will generate an ordering of the exams which for some reason cannot guide the constructive algorithm towards a good solution. In the case of a bad ordering of exams such as this, many of the exams cannot be scheduled without reshuffling exams that have already been scheduled earlier.

In Table 8, it can be observed that the SD heuristic ordering, the *Fuzzy SD+LE Model* and the *Fuzzy SD+LD Model* often produced solutions without invoking the *rescheduling procedure*. An interesting point here is that, although the SD heuristic ordering is capable of generating an ordering of exams that required no *rescheduling procedure*, when compared against the other single heuristic orderings it only produced three best results out of the 12 data sets (see Table 6). In contrast, the exam ordering generated using the *Fuzzy SD+LE Model* guides the constructive algorithm without requiring the *rescheduling procedure* and, moreover, it finds solutions which are better than other heuristics in 10 out of the 12 data sets. In addition, although the *Fuzzy SD+LE Model* needed to reschedule one exam in the case of *HEC-S-92-I* and *UTE-S-92*, the solutions were produced by performing only one iteration of the *rescheduling procedure*. For the same *HEC-S-92-I* data set, the SD heuristic ordering also produced only one skipped exam but it required 39 iterations, *on average*, of the *rescheduling procedure* to produce the solution. When the *UTE-S-92* data set is considered, although having only one unscheduled exam, an average of nine iterations of the *rescheduling procedure* were required to produce the solution.

Taking these facts into consideration, we may now speculate as to what might be the factors that cause the *Fuzzy SD+LE Model* to perform uniformly well across the 12 different data sets. Amongst the single heuristic orderings, LE performed well in eight out of 12 data sets (see Table 6), while SD often managed to find solutions without skipping an exam (see column 4 of Table 8). It would appear that the fuzzy approach is somehow combining the individual strengths of these two heuristics to improve the overall performance of the search algorithm.

In can be seen that 24 exams were skipped when the single heuristic ordering LD and the *Fuzzy SD+LD Model* were applied to the *STA-F-83-I* data set (columns 2 and 7 of Table 8). Interestingly, all these skipped exams are then scheduled by performing the *rescheduling procedure* with the same number of iterations, i.e. 24 (see columns 3 and 8 of Table 9). This means that none of the already scheduled exams needed to be bumped back to the unscheduled list in order to create spaces for the skipped exams. Further investigation has shown that the 24 skipped exams are the same in each case. We examined this closely in order to understand what had caused this curious effect.

In essence, the initial part of the construction process is a greedy algorithm that minimises the penalty of placing each exam, one by one, into the timetable (in the order given by the heuristic determination of difficulty). With the tendency to assign each unscheduled exam into the time slot with least penalty cost, the available time slots are usually occupied at an early stage of the scheduling process. In the case of the *STA-F-83-I* data set with the *Fuzzy SD+LD Model*, the first 13 exams were assigned to the 13 time slots available, although some of these exams could have been scheduled together in the same time slot—i.e. these 13 did *not* necessarily clash with each other. In effect, this situation had caused a 'bottleneck', after which no more valid time slots were available. In the next step of the construction process, the *rescheduling procedure* attempts to schedule each of the skipped exams by considering multiple simultaneous moves of already placed exams in order to obtain feasible solutions. For the *STA-F-83-I* data set, each of the skipped exams could be placed without needing to 'unschedule' ('bump-back') any exams already placed.

Turning now to the computational time, it seems that the *Fuzzy LD+LE Model* can be considered to be the best amongst the multiple heuristic orderings we experimented with since this heuristic always found good quality solutions in relatively low computational time. As seen in Table 6, in terms of solution quality, the *Fuzzy LD+LE Model* and *Fuzzy SD+LE Model* were approximately the same. Furthermore, when compared to the various single heuristic orderings, it is apparent that the *Fuzzy LD+LE Model* heuristic ordering obtained the minimum penalty cost for nine out of 12 data sets. However, in terms of computational time (see Table 10), the *Fuzzy SD+LE Model* and the *Fuzzy SD+LD Model* consistently perform worse than the other heuristic orderings. Note that the times in column 6 of Table 10 are equivalent to those for our previous approach [16].

Considering that the *Fuzzy LD+LE Model* combines two single heuristic orderings which are both categorised as *static* heuristics, it might be expected that this fuzzy model will take more computational time to produce the solution than the two heuristics on which it depends. However, the results presented in Table 10 indicate that in at least 6 out of the 12 cases the *Fuzzy LD+LE Model* is actually quicker than the single heuristics, specifically for the *CAR-F-92-I*, *CAR-S-91-I*, *LSE-F-91*, *RYE-F-92*, *TRE-S-92*, and *UTA-S-92-I* data sets. (It is arguable that is it also quicker for a 7th data set, *HEC-S-92-I*, as the *Fuzzy LD+LE Model* has a lower average than the other heuristics.) It can be seen that this fuzzy heuristic ordering always obtains the solutions in shorter execution time for the data sets that consist of more than 300 exams, except for *KFU-S-93*. For the rest of the data sets, the time taken to construct the solution is very reasonable compared to the other single static heuristics.

If we now compare the *longest* time required to produce the solutions among the static heuristic orderings (i.e. those excluding the use of SD) it is evident that the *Fuzzy LD+LE Model* always produced the solutions in relatively short time (except for *KFU-S-93*). This is obvious for the data sets that contain more than 300 exams particularly for *CAR-F-92-I*, *CAR-S-91-I* and *UTA-S-92-I* (see Table 10). For example, in the case of the *CAR-F-92-I* data set (looking at the *Worst* row), the *Fuzzy LD+LE Model* only took approximately 3 seconds, whereas the other heuristics took at least 217 seconds. Although it takes a long time to search for the 'best' fuzzy model, it is important to notice how quickly the 'best' fuzzy model finds the solution compared to these other heuristic orderings.

However, the capability to produce solutions quickly is not achievable when the dynamic heuristic is implemented. As seen in Table 10, the *Fuzzy SD+LD Model* required the longest time in all problem instances as compared to the other heuristics, followed by the *Fuzzy SD+LE Model*. We believe that most of the time is used to recalculate the number of valid time slots available for the remainder of the unscheduled exams, and not to calculate the fuzzy exam weight. This assumption is based on the observation mentioned earlier, that the *Fuzzy LD+LE Model* always obtained the solutions in quick time, meaning that the time taken to calculate exam fuzzy weight must be relatively very small. Moreover, in 10 out of the 12 problem instances, the *Fuzzy SD+LE Model* found the solutions without invoking the *rescheduling procedure* (and the other two data sets with only one iteration of the *rescheduling procedure*), which means that no time was spent reshuffling the scheduled exams.

8. Conclusions

The work presented in this paper builds upon and extends our previous work investigating the use of fuzzy techniques in the construction of university examination timetables. In our previous work [16], we carried out a preliminary investigation into the use of fuzzy techniques to combine multiple heuristics used to determine the order in which to place exams into timetables in the construction phase. This previous work considered only a limited combination of heuristics but, nevertheless, highlighted the fact that the fuzzy approach showed promise. In this paper, we have extended this by considering the fuzzy combination of all three pairs of heuristics that are possible from a set of three single heuristics (LD, LE and SD). We have also analysed, in significant depth, the effect of utilising the fuzzy combination of heuristics in the construction process, specifically in its effect on the amount of backtracking required and the associated computational time required. We have evaluated the proposed approach on the 12 Carter benchmark data sets commonly used for comparative purposes in this field of research.

We have shown that the fuzzy combination of SD and LE obtained a good overall performance in terms of low penalty cost on the 12 benchmark data sets. The use of the fuzzy combination of heuristics resulted in lower penalty costs than obtained by the use of any of the heuristics alone, on all 12 data sets. Furthermore, the *Fuzzy SD+LE Model* obtained a penalty cost for one data set (*YOR-F-83-I*) that is lower than any other previously published constructive approach. In addition to this, the *Fuzzy SD+LE Model* required less backtracking in the construction process than that observed for any single heuristic used alone. We hence conclude that this novel fuzzy approach is a highly effective method for combining heuristics to be used in the construction of examination timetabling solutions. As the process of constructing feasible solutions to examination timetables in real-world situations (as opposed to these benchmark data sets) is often a significant challenge in its own right [43], the fuzzy approach detailed here could be of great benefit.

The issue of which combination of heuristics should be applied in novel real-world problem instances is an important open research question. Our current research is investigating the development of a framework which could take as input a new problem instance and, from its characteristics, recommend which combination of heuristics should be used for producing a solution.

Acknowledgements

This research work was supported by the Universiti Teknologi Malaysia (UTM), the Ministry of Science, Technology and Innovation (MOSTI) Malaysia and the Engineering and Physical Sciences Research Council (EPSRC).

References

[1] De Werra D. An introduction to timetabling. European Journal of Operational Research 1985;19:151–62.
[2] Burke EK, Newall JP. A multi-stage evolutionary algorithm for the timetable problem. IEEE Transactions on Evolutionary Computation. 1999; 3.1: 63–74.
[3] Burke EK, Newall JP, Weare RF. A memetic algorithm for university exam timetabling. In: Burke EK, Ross P, editors. Practice and theory of automated timetabling, first international conference, Edinburgh, U.K., August 29–September 1, 1995, Selected papers, Lecture notes in computer science, vol. 1153, Berlin: Springer; 1996. p. 241–50.
[4] Deris S, Omatu S, Ohta H, Saad P. Incorporating constraint propagation in genetic algorithm for university timetabling planning. Engineering Applications of Artificial Intelligence 1999;12:241–53.
[5] Ueda H, Ouchi D, Takahashi K, Miyahara T. Comparisons of genetic algorithms for timetabling problems. Systems and Computers in Japan 2004;35(7):1–12 [translated from Denshi Joho Tsushin Gakkai Ronbunshi, vol. J86-D-I, No. 9, September 2003. p. 691–701].
[6] Di Gaspero L, Schaerf A. Tabu search techniques for examination timetabling. In: Practice and Theory of Automated Timetabling III (PATAT 2000, Konstanz Germany, August, selected papers), Lecture notes in computer science, vol. 2079, Springer, Berlin Heidelberg, New York: 2001. p. 104–17.
[7] Burke EK, Kendall G, Soubeiga E. A tabu-search hyperheuristic for timetabling and rostering. Journal of Heuristics 2003;9(6):451–70.
[8] White GM, Xie BS, Zonjic S. Using tabu search with longer-term memory and relaxation to create examination timetables. European Journal of Operational Research 2004;153:80–91.
[9] Kendall G, Hussin NM. A tabu search hyper-heuristic approach to the examination timetabling problem at the MARA university of technology. In: Practice and theory of automated timetabling V, fifth international conference, PATAT 2004, Pittsburgh, PA, USA, August 18–20, 2004, Revised selected papers, Lecture notes in computer science, vol. 3616. Berlin: Springer; 2005. p. 270–93.
[10] Thompson JM, Dowsland KA. A robust simulated annealing based examination timetabling system. Computers & Operations Reasearch 1998;25(7/8):637–48.
[11] Guéret C, Jussien N, Boizumault P, Prins C. Building university timetables using constraint logic programming, In: Practice and theory of automated timetabling, first international conference, Edinburgh, U.K., August 29–September 1, 1995, Selected papers, Lecture notes in computer science, vol. 1153, Berlin: Springer; 1996. p. 130–45.
[12] Boizumault P, Delon Y, Peridy L. Constraint logic programming for examination timetabling. The Journal of Logic Programming 1996;26(2): 217–33.
[13] Abdennadher S, Marte M. University course timetabling using constraint handling rules. Journal of Applied Artificial Intelligence 2000;14(4):311–26.
[14] Yang Y, Petrovic S. A novel similarity measure for heuristic selection in examination timetabling. In: Practice and theory of automated timetabling V, fifth international conference, PATAT 2004, Pittsburgh, PA, USA, August 18–20, 2004, Revised selected papers, Lecture notes in computer science, vol. 3616, Berlin: Springer; 2005.
[15] Burke EK, Petrovic S, Qu R. Case based heuristic selection for timetabling problems. Journal of Scheduling 2006;9(2):115–32.
[16] Asmuni H, Burke EK, Garibaldi JM, McCollum B. Fuzzy multiple heurisitc orderings for examination timetabling. In: Practice and theory of automated timetabling V, fifth international conference, PATAT 2004, Pittsburgh, PA, USA, August 18–20, 2004, Revised selected papers, Lecture notes in computer science, vol. 3616, Berlin: Springer; 2005. p. 334–53.
[17] Petrovic S, Patel V, Yang Y. University timetabling with fuzzy constraints. In: Practice and theory of automated timetabling V, fifth international conference, PATAT 2004, Pittsburgh, PA, USA, August 18–20, 2004, Revised selected papers, Lecture notes in computer science, vol. 3616, Berlin: Springer; 2005.
[18] Carter MW. A survey of practical applications of examination timetabling algorithms. Operation Research 1986;34(2):193–202.
[19] Bardadym VA. Computer aided school and university timetabling: the new wave. In: Practice and theory of automated timetabling, first international conference, Edinburgh, U.K., August 29–September 1, 1995, Selected papers, Lecture notes in computer science, vol. 1153, Berlin: Springer; 1996. p. 22–45.
[20] Carter MW, Laporte G. Recent development in practical examination timetabling. In: Practice and theory of automated timetabling, first international conference, Edinburgh, U.K., August 29–September 1, 1995, Selected papers, Lecture notes in computer science, vol. 1153, Berlin: Springer; 1996. p. 3–21.
[21] Burke EK, Jackson K, Kingston JH, Weare RF. Automated university timetabling: the state of the art. The Computer Journal 1997;40(9): 565–71.
[22] Schaerf A. A survey of automated timetabling. Artificial Intelligent Review 1999;13:87–127.
[23] Burke EK, Petrovic S. Recent research directions in automated timetabling. European Journal of Operational Research 2002;140:266–80.
[24] Petrovic S, Burke EK. University timeatbling, Handbook of scheduling: algorithms, models, and performance analysis. Boca Raton, FL: CRC Press; 2004 [Chapter 45].
[25] Burke EK, Elliman DG, Ford PH, Weare RF. Examination timetabling in british universities—a survey. In: Practice and theory of automated timetabling, first international conference, Edinburgh, U.K., August 29–September 1, 1995, Selected papers, Lecture notes in computer science, vol. 1153, Berlin: Springer; 1996. p. 76–90.

[26] Carter MW, Laporte GG, Lee SY. Examination timetabling: algorithmic strategies and applications. Journal of the Operational Research Society 1996;47:373–83.
[27] Burke EK, Bykov Y, Newall J, Petrovic S. A time-predefined local search approach to exam timetabling problems. IIE Transactions 2004;36(6):509–28.
[28] Broder S. Final examination scheduling. Communications of the ACM 1964;7:494–8.
[29] Cole AJ. The preparation of examination time-tables using a small-store computer. The Computer Journal 1964;7(2):117–21.
[30] Foxley E, Lockyer K. The construction of examination timetables by computer. The Computer Journal 1968;11(3):264–8.
[31] Burke EK, Kendall G, edtiors. Search methodologies—introductory tutorials in optimization and decision support techniques. Berlin: Springer; 2005.
[32] Dueck G. New optimization heuristics: the great deluge algorithm and the record-to-record travel. Journal of Computational Physics 1993;104: 86–92.
[33] Johnson D. Timetabling university examinations. Journal of Operational Research Society 1990;41(1):39–47.
[34] Zadeh LA. Fuzzy sets. Information and Control 1965;8:338–53.
[35] Cox E, O'Hagen M. The fuzzy systems handbook: a practitioner's guide to building, using and maintaining fuzzy systems. Cambridge, MA: AP Professional; 1998.
[36] Zimmerman HJ. Fuzzy set theory and its applications. 3rd ed., Dordrecht: Kluwer Academic Publishers; 1996.
[37] Garibaldi JM. Fuzzy expert systems. In: Gabrys B, Leiviska K, Strackeljan J, editors. Do smart adaptive systems exist? Best practice for selection and combination of intelligent methods. Berlin: Springer; 2005. p. 105–32.
[38] Lim MH, Rahardja S, Gwee BH. A GA paradigm for learning fuzzy rules. Fuzzy Sets and Systems 1996;82:177–86.
[39] Burke EK, Newall JP. Solving examination timetabling problems through adaptation of heuristic orderings. Annals of Operations Research 2004;129:107–34.
[40] Burke EK, Meisels A, Petrovic S, Qu R. A graph-based hyper heuristic for educational timetabling problems. European Journal of Operational Research 2007;176:177–92.
[41] Caramia M, DellOlmo P, Italiano GF. New algorithms for examination timetabling. In: Naher S, Wagner D, editors. Algorithm engineering 4th international workshop, Proceedings of WAE 2000 (Saarbrucken, Germany, September), Lecture notes in computer science, vol. 1982, Berlin: Springer; 2001. p. 230–41.
[42] Burke EK, Eckersky, AJ, McCollum B, Petrovic S, Qu R. Hybrid variable neighbourhood approaches to university examination timetabling, computer science technical report, NOTTCS-TR-2006-2, University of Nottingham, 2006.
[43] McCollum B. A perspective on bridging the gap between theory and practice in university timetabeling. Practice and theory of automated timetabeling VI. Revised selected paper, 6th international conference, Bono Springs Lecture Notes in Computer Science, vol. 3867, August/September 2006, p. 3–23.
[44] Qu R, Burke EK, McCollum B, Merlok LT, L Lee Sy. A survey of search methodologies and automated approaches for examination timetabling. Journal of Scheduling, 2008, to appear.
[45] Burke EK, Kingston J, de Werra D. In:Applications to timetabling section 5.6 of the handbook of graph theory. Gross J, Yellen J, editors, Chapmon Hall ICRC Press, 2004. p. 445–74.