A Tabu-based Memetic Approach to the Examination Timetabling Problem

Salwani Abdullah[1], Hamza Turabieh[1], Barry McCollum[2]

[1]Center for Artificial Intelligence Technology,
Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia
{salwani, hamza}@ftsm.ukm.my
[2]Department of Computer Science, Queen's University Belfast, Belfast BT7 1NN
United Kingdom
b.mccollum@qub.ac.uk

Abstract.

Constructing examination timetable for higher educational institutions is a very complex work due to the complexity of timetabling problems. The objective of examination timetabling problem is to satisfy the hard constraints and minimize the violations of soft constraints. In this paper, a tabu-based memetic approach has been applied and evaluated against the latest methodologies in the literature on standard benchmark problems. The approach which hybridizes a concept of tabu list and memetic algorithm. The tabu list is used to penalise neighbourhood structures that are unable to generate better solutions after the crossover and mutation operators being applied to the selected solutions from the population pool. We demonstrate that our approach is able to enhance the quality of the solutions by carefully selecting the effective neighbourhood structures. Hence, some best known results have been obtained.

1 Introduction

Timetabling problems have long been a challenging area for researchers across both Operational Research and Artificial Intelligence. Among the wide variety of timetabling problems, educational timetabling (which cover areas such as school, course and exam timetabling) is one of the most widely studied. This concentrates on examination timetabling problems which can be generally defined as assigning a set of exams into a limited number of timeslots and rooms subject to a set of constraints.

In the past, a wide variety of approaches for solving the examination timetable problem have been described and discussed in the literature. For a recent detailed overview readers should consult Qu et al (2009). Carter (1986) categorised the approaches taken into four types: sequential methods, cluster methods, constraint-based methods and generalised search. Petrovic and Burke (2004) added the following

categories: hybrid evolutionary algorithms, meta-heuristics, multi-criteria approaches, case based reasoning techniques, hyper-heuristics and adaptive approaches.
These approaches are tested on various examination timetabling datasets that can be found from http://www.asap.cs.nott.ac.uk/resource/data. Due to the existence of multi formulations for university timetabling problem, the 2nd International Timetabling Competition (ITC2007) (McCollum et al. 2009) attempted to standardize the problems found within educational timetabling by introducing three tracks (one on exam and two on course timetabling) where the problems incorporated more real-world constraints. In doing so, the organizers attempted to reduce the acknowledged gap between research and practice which exists in this area (McCollum 2007). Interested readers can find more details about examination timetabling research in the comprehensive survey paper by Qu et al. (2009) and Lewis (2008).

The rest of the paper is organised as follows. The next section formally discusses the basic algorithm of memetic approach. Section3 presents the examination timetabling problem and formulation. The solution approach is outlined in Section 4. Our results are presented, discussed and evaluated in Section 5. This is followed by some brief concluding comments in Section 6.

2 Memetic Approach

Memetic algorithms which is a population-based approach were proposed by Moscato and Norman in 1992 are an extension of genetic algorithms which sometimes is called hybrid genetic algorithms or genetic local search algorithms (Hart et al. 2004). The idea is that individuals within a population can be improved within a generation that can be done by employing local search methods on individual members of a population between generations.

Burke et al. (1996) employed a memetic algorithm for university examination timetabling where two evolutionary operators are used (light and heavy mutation) in the initial phase followed by a hill-climbing algorithm. The algorithm has been tested on real examination datasets. Experimental results show that the solution quality found was better when compared to employing evolutionary operators alone. The effects of diversity in initial populations in memetic algorithms has been investigated by the same authors (see Burke et al. 1996). The technique by Paechter et al. (1996) implemented an extension of memetic algorithm to the lecture timetabling problem which utilised several types of mutation strategies. Experimental results show that *selfish* and *co-operative* mutations are very useful in increasing the performance of the algorithm when applied to this problem. Burke et al. (1998) introduced randomness (that used three diversity measures) in the initial population to generate a high level of diversity. The study of diversity in initialisation has shown great potential benefits for memetic algorithms. Burke and Newall (1999) applied a hybrid method of heuristic sequencing and evolutionary methods to the examination timetabling problem naming the approach as a multi-stage evolutionary algorithm. In this approach, the hybrid method is applied to a subset of events (examinations) at a particular time. The algorithm then fixes the events in the timetable before moving to the next subsets (which is like a decomposition process). In order to evaluate the effectiveness of this approach, real datasets were used. The results show that this approach was able to improve the solution quality and reduce the time taken to find that solution. A number of relevant issues on the design of memetic algorithm for scheduling and timetabling problems can be seen in Burke and Landa Silva (2004). A number of studies on memetic algorithms on various combinatorial optimisation problems can be found in Krosnogor and Smith (2005) and Osman and Laporte (1996). Interested readers can find more details about similar approaches in Moscato (1999, 2002). Other related papers on population-based approach applied to examination timetabling problems can be found in Eley (2007) and Ersoy et al. (2007).

3 Problem Description

The examination timetabling problem is a command problem found in schools and higher educational institutes which are concerned with allocating exams into a limited number of timeslots (periods) subject to a set of constraints (see Burke et al. 1996). There are generally two categories of constraints: hard and soft. Hard constraints must be completely satisfied and cannot be violated. Examples of generally accepted hard constraints are:

- no student can sit in two exams simultaneously
- the scheduled exams must not exceed the room capacity

In addition, often hard constraints exist relating to the ordering of examination. Solutions that satisfy all hard constraints are often called *feasible* solutions. Soft constraints, which are desirable and not essential i.e. their satisfaction dictates the overall quality of the gained solution. A particularly common soft constraint refers to spreading exams as evenly as possible over the schedule as discussed in Burke et al. (1996). ITC2007 presented to the research community an extensive listing of additional soft constraints. In real-world situations, it is usually impossible to satisfy all soft constraints, but minimising the violations of soft constraints represents an increase in the quality of the solution.

The problem description that is employed in this paper is adapted from the description presented in Burke et al. (2004). The input for the examination timetabling problem can be stated as follows

- E_i is a collection of N examinations (i=1,…,N).
- P is the number of timeslots.
- $C=(c_{ij})_{NxN}$ is the conflict matrix where each record, denoted by c_{ij} ($i,j \in \{1,\ldots,N\}$) ,represent the number of students taking the exams i and j.
- M is the number of students.
- t_k ($1 \le t_k \le T$) specifies the assigned timeslots for exam k ($k \in \{1,\ldots,N\}$.

The following hard constraints are considered in this paper:

- no students should be required to sit two examinations simultaneously.
- examinations with common students should be scheduled in different periods.

In this problem, we formulate an objective function which tries to spread out students' exams throughout the exam period (Expression (1)).

$$\min \frac{\sum_{i=1}^{N-1} F(i)}{M} \tag{1}$$

where

$$F(i) = \sum_{j=i+1}^{N} c_{ij} . proximity(t_i, t_j) \tag{2}$$

$$proximity(t_i, t_j) = \begin{cases} 2^5 / 2^{|t_i - t_j|} & if\ 1 \le |t_i - t_j| \le 5 \\ 0 & otherwise \end{cases} \tag{3}$$

subject to:

$$\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{ij} . \lambda(t_i, t_j) = 0 \quad \text{where} \quad \lambda(t_i, t_j) = \begin{cases} 1 & if\ t_i = t_j \\ 0 & otherwise \end{cases} \tag{4}$$

Equation (2) presents the cost for an exam *i* which is given by the proximity value multiplied by the number of students in conflict. Equation (3) represents a proximity value between two exams (Carter et al. 1996). Equation (4) represents a clash-free requirement so that no student is asked to sit two exams at the same time. The clash-free requirement is considered to be a hard constraint.

4 The Tabu-based Memetic Approach

The approach described here consists of a construction stage followed by improvement.

4.1 Construction Heuristic

A construction algorithm which is based on a saturation degree graph colouring heuristic is used to generate large populations of feasible timetables. The approach starts with an empty timetable. The exams with highest number of exams in conflict and more likely to be difficult to be scheduled will be attempted first without taking into consideration the violation of any soft constraints, until the hard constraints are met. More details on graph colouring applications to timetabling can be found in Burke et al. (2004).

4.2 Improvement Algorithm

In this paper, a tabu-based memetic approach is proposed as an improvement algorithm that operates on a set of possible solutions (generated from the construction heuristic) to solve examination timetabling problem, where a set of neighbourhood structures (as discussed in subsection 4.3) have been used as a local search mechanism. The aim of using a set of neighbourhood structure inside genetic operators is to produce significant improvements in a solution quality. In this approach, a concept of tabu list is employed to penalise neighbourhood structures that are unable to generate better solutions after the crossover and mutation operators being applied to the selected solutions from the population pool. The schematic overview of the algorithm is presented in Fig. 1 where the populations are generated using the construction heuristic. Two parents (solutions) will be selected from the population based on roulette wheel procedure. Two memetic operators i.e. crossover and mutation will be employed prior to the improvement algorithm (i.e. tabu-based memetic algorithm). The best solution found after the employment of the improvement algorithm will be added to the population pool while maintaining the size of the population.

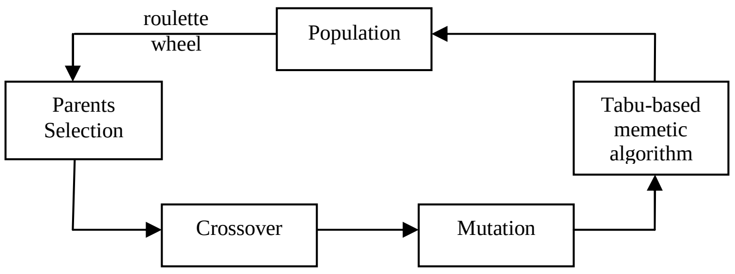


Fig. 1. A schematic overview of the approach

The pseudo code of the algorithm is presented in Fig. 2. The algorithm begins by creating initial populations. A best solution from the population is selected, S_{best}. A tabu list with the size *TabuSize* is created with an aim to hold ineffective neighbourhood structures from being selected in the next iteration and will give more chances for the remaining neighbourhood structures to be explored (see Table 3 for parameter setting). A variable *CanSelect*, (represents a boolean value) that allows the algorithm to control the selection of a neighbourhood structure. For example if a neighbourhood structures, *Nbs*, shows a good performance (in terms of producing a lower penalty solution), so the algorithm will continue uses *Nbs* in the next iterations 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. In a *do-while* loop, two solution are randomly selected, S_1 and S_2. The crossover and mutation operators are applied on S_1 and S_2 to obtain S_1' and S_2'. Randomly select a neighbourhood structure, *Nbs*, and applied on 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 an assigned to a current solution S^*. If S^* is better than the best solution in hand, S_{best}, then S^* is accepted. Otherwise *Nbs* will be added into the T_{List}. The member of the populations 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 and to be used in the next generation. The process is repeated and stops when the termination criterion is met (in this work the termination criterion is set as a computational times).

```
for $i$=1 to population size
    Generate random solutions, $S_i$;
end
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
        Randomly 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 $S_2$";
    Get 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 good and worse
    solutions, respectively, while maintaining the size of the population;
end while
Output the best solution, $S_{best}$;
```

Fig.2 Tabu-based memetic algorithm

4.3 Neighbourhood Structures

Ahuja et al. (2000) in their paper highlighted the importance of the neighbourhood structure in the local or neighbourhood search. They said,

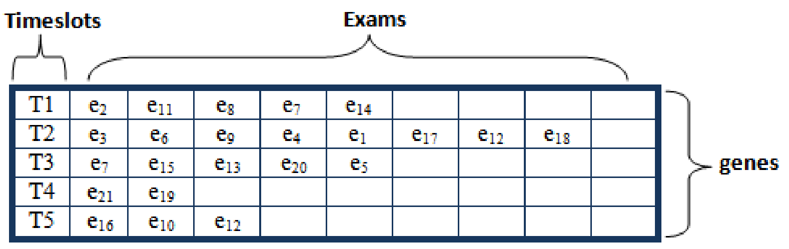
> "*A critical issue in the design of a neighbourhood search approach is the choice of the neighbourhood structure that is the manner in which the neighbourhood is defined.*"

Some techniques in the literature, like simulated annealing and tabu search, generally use a single neighbourhood structure throughout the search and focus more on the parameters that affect the acceptance of the moves rather than the neighbourhood structure. Thompson and Dowsland (1996, 1998) discussed how the choice of the neighbourhood structure affects the quality of solutions obtained for examination timetabling. We can say that the success of finding good solutions for these problems is determined by the technique itself and the neighbourhood structure employed during the search. In this paper, a set of different neighbourhood structures have been employed. Their explanation can be outlined as follows (adapted from Abdullah et al. 2007):

Nbs$_1$: Select two exams at random and swap timeslots.
Nbs$_2$: Choose a single exam at random and move to a new random feasible timeslots.
Nbs$_3$: Select two timeslots at random and simply swap all the exams in one timeslots with all the exams in the other timeslots.
Nbs$_4$: 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 exams that 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.
Nbs$_5$: Move the highest penalty exams from a random 10% selection of the exams to a random feasible timeslots.
Nbs$_6$: Carry out the same process as in Nbs$_5$ but with 20% of the exams.
Nbs$_7$: Move the highest penalty exams from a random 10% selection of the exams to a new feasible timeslots which can generate the lowest penalty cost.
Nbs$_8$: Carry out the same process as in Nbs$_7$ but with 20% of the exams
Nbs$_9$: Select two timeslots based on the maximum enrolled exams, say t_i and t_j .Select the most conflicted exam in t_i with t_j and then apply Kempe chain from Thompson and Dowsland (1996).

4.4 Solution Representation

A direct representation is used. Each population member (which represents a feasible solution) is represented as a number of genes that contain information about the timeslot and exams. For example e_2, e_{11}, e_8, e_7, e_{14} are scheduled at timeslot T_1.



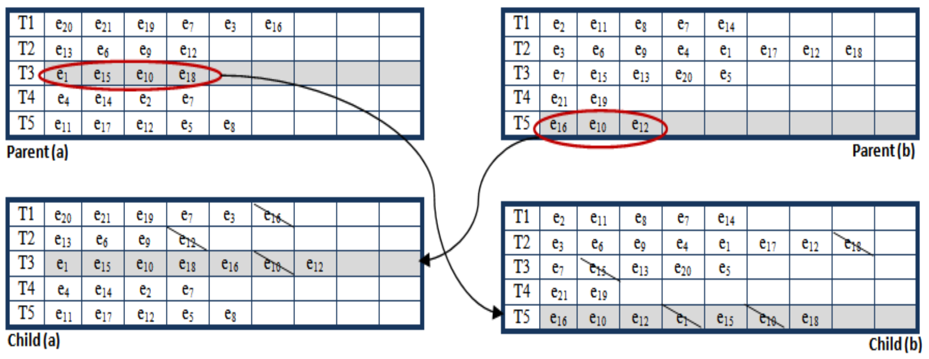
T1	e_2	e_{11}	e_8	e_7	e_{14}			
T2	e_3	e_6	e_9	e_4	e_1	e_{17}	e_{12}	e_{18}
T3	e_7	e_{15}	e_{13}	e_{20}	e_5			
T4	e_{21}	e_{19}						
T5	e_{16}	e_{10}	e_{12}					

Fig.3 Solution representation

4.5 The Memetic Operators: Crossover and Mutation

There are different types of crossover operators available in the literature. For example Cheong et al (2007) applied a crossover operation based on the best days (three periods per day and minimum number of clashes per students). In this paper, we applied a period exchange crossover. This crossover operator allows a number of exams (from one timeslot) to be added to another timeslot and vice versa based on a crossover rate. The repair mechanism is applied to ensure the feasibility of the child. The crossover operation is illustrated in Fig.4, represents a period-exchange crossover. The shaded periods represent the selected periods for a crossover operation. These periods are selected based on a crossover rate using roulette wheel selection method, for example timeslots T_3 and T_5 are chosen as parents (a) and (b), respectively. Crossover is performed by inserting all exams from timeslot T_5 in parent (b) to timeslot T_3 in parent (a), which then will produce 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 exams. For example; in child (a), e_{10} is repeated in T_3 (occurs twice), which should be removed; e_{16} and e_{12} that are located in T_1 and $T_{2,}$ respectively, should also be removed to insure that child (a) is feasible. In child (b), e_1 is conflicted with e_{16}, as a result e_1 cannot be inserted in T_5, while e_{10} and e_{18} are occur twice in T_5 and T_2, respectively, so these exams should be removed to obtain feasibility. Removing conflicts and repeating exams in each timeslot is considered as a repair function that changed the infeasibility of each offspring to feasible once.

The mutation is used to enhance the performance of crossover operation in allowing a large search space to be explored. Random selection of neighbourhood structures is used in a mutation process based on a mutation rate obtained after preliminary experiments (see Table 2 for a parameter setting).



Parent (a)

T1	e_{20}	e_{21}	e_{19}	e_7	e_3	e_{16}			
T2	e_{13}	e_6	e_9	e_{12}					
T3	e_1	e_{15}	e_{10}	e_{18}					
T4	e_4	e_{14}	e_2	e_7					
T5	e_{11}	e_{17}	e_{12}	e_5	e_8				

Parent (b)

T1	e_2	e_{11}	e_8	e_7	e_{14}				
T2	e_3	e_6	e_9	e_4	e_1	e_{17}	e_{12}	e_{18}	
T3	e_7	e_{15}	e_{13}	e_{20}	e_5				
T4	e_{21}	e_{19}							
T5	e_{16}	e_{10}	e_{12}						

Child (a)

T1	e_{20}	e_{21}	e_{19}	e_7	e_3	~~e_{16}~~			
T2	e_{13}	e_6	e_9	~~e_{12}~~					
T3	e_1	e_{15}	e_{10}	e_{18}	e_{16}	~~e_{10}~~	e_{12}		
T4	e_4	e_{14}	e_2	e_7					
T5	e_{11}	e_{17}	e_{12}	e_5	e_8				

Child (b)

T1	e_2	e_{11}	e_8	e_7	e_{14}				
T2	e_3	e_6	e_9	e_4	e_1	e_{17}	e_{12}	~~e_{18}~~	
T3	e_7	~~e_{15}~~	e_{13}	e_{20}	e_5				
T4	e_{21}	e_{19}							
T5	e_{16}	e_{10}	e_{12}	~~e_1~~	e_{15}	~~e_{10}~~	e_{18}		

Fig.4 Chromosome representation after crossover

5 Experimental Results

The proposed algorithm was programmed using Matlab and simulations were performed on the Intel Pentium 4 2.33 GHz computer.

5.1 Problem instances

In this paper, we considered on a standard benchmark examination timetabling problem from Carter et al. (1996) as shown in Table 1. In this work, we evaluate the performance of our approach on eleven instances. We intend comparing our technique to the substantial body of work which has been published since the release of the Carter datasets. This will allow us to understand just how effective our technique is. Once we have achieved this, we intend looking at ITC2007 datasets (see McCollum et al. 2007). To look at ITC2007 first would provide us with limited scope in relation to understanding how good our technique is as very few papers have been published so far. We do intend doing this in future work.

The parameters used in the algorithm are chosen after preliminary experiments as shown in Table 2 (and are comparable similar with the papers in Abdullah and Turabieh, 2008). In this approach, we reduced the population size (compared to the one in Abdullah and Turabieh, 2008) with an aim to speed up the searching process and to reduce the time taken to generate the populations.

Instance	Number of periods	Number of examination	Number of Students	Density of Conflict Matrix
car-s-91	35	682	16925	0.14
car-f-92	32	543	18419	0.13
ear-f-83	24	190	1125	0.29
hec-s-92	18	81	2823	0.42
kfu-s-93	20	461	5349	0.06
lse-f-91	18	381	2726	0.06
sta-f-83	13	139	611	0.07
tre-s-92	23	261	4360	0.14
uta-s-92	35	622	21267	0.18
ute-s-92	10	184	2750	0.13
yor-f-83	21	181	941	0.08

Table 1 Examination timetabling datasets

Parameter	Value
Population size	50
Crossover rate	0.8
Mutation rate	0.04
Selection mechanism	Roulette wheel selection

Table 2 Parameter setting

5.2 Preliminary Experiments

The preliminary experiments have been performed to test the most appropriate tabu size to be used in the improvement algorithm as discussed in subsection 4.2. Six different sizes of the tabu list have been tested (tabu size = 1 to 6) and one without tabu (tabu size = 0). The algorithm was tested on three datasets. Table 3 shows the average of 5 runs with different tabu sizes with each run takes about 5 to 8 hours. It shows that, the algorithm is able to obtain better results with tabu size = 3.

Tabu size	Sat-f-83	Kfu-s-93	Ute-s-92
0	159.53	14.05	26.48
1	159.27	13.53	26.35
2	158.59	13.25	**25.03**
3	**157.76**	**13.00**	25.18
4	158.23	13.40	25.45
5	158.64	13.2	25.64

Table 3 Performance comparison based on different tabu sizes

This is believed that the higher the value of tabu size, the longer the neighbourhood structures will remain in the tabu list. This limits the search space. We notice that a higher value of tabu size makes the solution considerably worse, since in this case less number of neigbourhood structures available to be employed in the next search process, and thus more difficult to improve. Note that in the next experiment, the tabu size = 3 is used to evaluate the performance of our algorithm (see subsection 4.3).

5.3 Comparison Results

We ran the experiments between 5 to 8 hours for each of the datasets. Other techniques reported here run their experiments between 5 to 9 hours. For example Yang and Petrovic (2005) took more than 6 hours and Cote et al. (2005) about 9 hours Note that running a system within these periods is not unreasonable in the context of examination timetabling where the timetables are usually produced months before the actual schedule is required. Table 4 provides the comparison of our results with the best known results for these benchmark datasets (taken from Qu et al. 2009). The best results out of 5 runs are shown in bold.

Instance	Our approach	Best known	Authors for best known
car-s-91	**4.35**	4.50	Yang and Petrovic (2005)
car-f-92	**3.82**	3.93	Yang and petrovic (2005)
ear-f-83	33.76	**29.3**	Caramia et al. (2001)
hec-s-92	10.29	**9.2**	Caramia et al. (2001)
kfu-s-93	**12.86**	13.0	Burke et al. (2006)
lse-f-91	10.23	**9.6**	Caramia et al. (2001)
sta-f-83	**155.98**	157.2	Cote et al. (2005)

tre-s-92	8.21	**7.9**	Burke et al. (2006)
uta-s-92	3.22	**3.14**	Yang and Petrovic (2005)
ute-s-92	25.41	**24.4**	Caramia et al. (2001)
yor-f-83	36.35	**36.2**	Caramia et al. (2002), Abdullah et al. (2007)

Table 4 Comparison results

Our algorithm produces better results on four out of eleven datasets. We are particularly interested to compare our results with the other results in the literature that employed population based algorithms i.e.: Cote et al. (2005) that employed bi-objective evolutionary algorithm with local search operators in the recombination process; Burke et al. (2006) that employed genetic algorithms on selecting subset of neighbourhood in variable neighbourhood search; Eley (2007) that applied ant algorithm with hill climbing operators and Ersoy et al. (2007) that applied memetic algorithm hyper-heuristics with three hill climbers chosen adaptively or deterministically.

Table 5 shows the comparison results on the population based algorithms as mentioned above. Again, the best results out of 5 runs are shown in bold. Note that the value marked "-" indicates that the corresponding problem is not tested. From Table 5, we can see that our algorithm produces better results on all datasets when compared against the method of Eley (2007) and Ersoy et al. (2007). Note that Ersoy et al. (2007) only attempt to solve six out of eleven datasets presented here. Our approach produces better results than Cote et al. (2005) and Burke et al. (2006) on eight and seven datasets, respectively. It is clearly shown that our tabu-based memetic approach out performs other population based algorithms on most of the instances. We believe that with the help of a tabu list, the algorithm performs better and able to find a better solution because the non-effective neighbourhood structures will not be employed in the next iterations i.e. the algorithm will only be feed with the currently effective neighborhood structure.

Instance	Our approach	Cote et al. (2005)	Burke et al. (2006)	Eley (2007)	Ersoy et al. (2007)
car-s-91	**4.35**	5.2	4.6	5.2	-
car-f-92	**3.82**	4.2	4.0	4.3	-
ear-f-83	**33.76**	34.2	37.92	36.8	-
hec-s-92	10.29	**10.2**	12.25	11.1	11.7
kfu-s-93	**12.86**	14.2	13.0	14.5	15.8
lse-f-91	10.23	11.2	**10.0**	11.3	13.3
sta-f-83	**155.98**	157.2	159.9	157.3	157.9
tre-s-92	8.21	8.2	**7.9**	8.6	-
uta-s-92	3.22	**3.2**	**3.2**	3.5	-
ute-s-92	25.41	25.2	**24.8**	26.4	26.7
yor-f-83	36.35	**36.2**	37.28	39.4	40.7

Table 5 Comparison results on population based algorithms

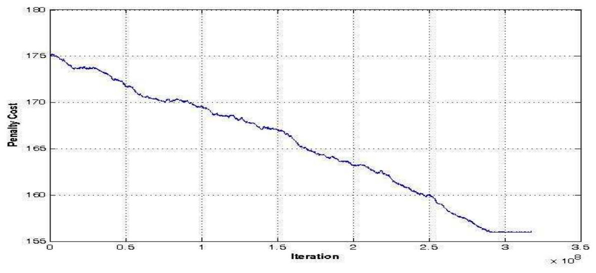


Fig. 4 *sta-f-83* convergence

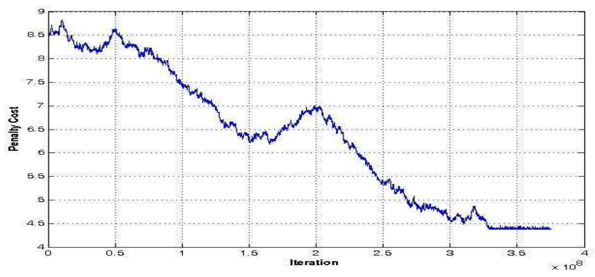


Fig. 5 *car-s-91* convergence

Fig. 4 and Fig. 5 show the behaviour of the algorithm when applied to two of the datasets i.e. *sta-f-83* and *car-s-91*, respectively. In all the figures, the x-axis represents the number of iterations while the y-axis represents the penalty cost. Every point in the graphs corresponds to the penalty cost and number of iterations of a separate solution. These graphs show how our algorithm explores the search space. In Fig. 4 the solution is steadily improved as the search time increases until it reaches a steady state towards the end of the search process. However, in Fig. 5, the curve that represents the quality of the current solution, moves up and down and slowly converge to a better solution. The difference of convergence process between these two datasets is believed to have a relation to the value of the conflict density. The higher matrix conflict density signifies that more exams are conflicting with each other. From Table 1, we can see that the conflict density value of *car-s-91* (0.14) is higher compared to the conflict density value of *sta-f-83* (0.07). This implies that we have less feasible solution points in our search space (for *car-s-91*). This is proven by the presented graph in Fig. 5 that shows the convergence process is not as smooth as on *sta-f-83* dataset.

Fig. 6 (a-e), show the box plots of the cost when solving *car-s-91*, *car-f-92*, *sta-f-83*, *ear-f-83*, and *kfu-s-93* instances, respectively.

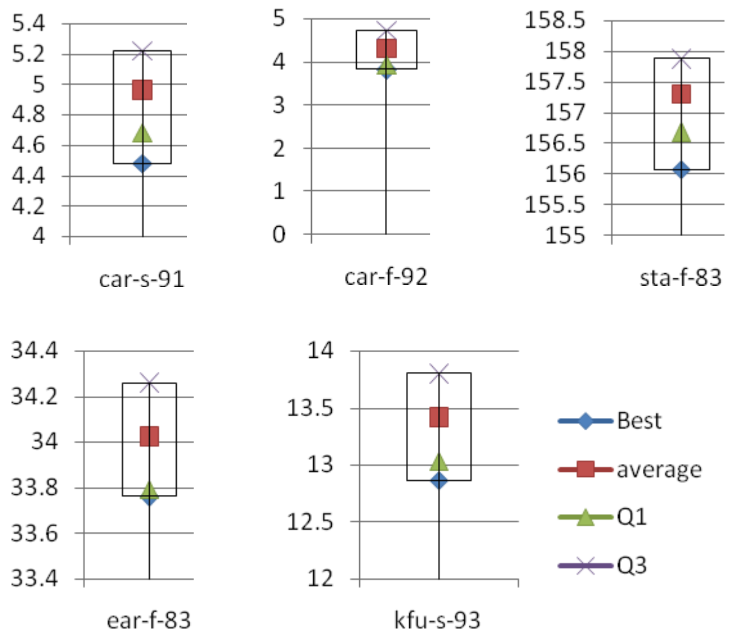


Fig. 6 (a-f). Box plots of the penalty costs for *car-s-91*, *car-f-92*, *sta-f-83*, *ear-f-83* and *kfu-s-93* instances, respectively

The results from the figures show less dispersions of solution points. We believe that by employing a series of neighbourhood structures and a tabu list to penalize unperformed neighbourhood structures to be used later within the search algorithm helps the search algorithm to differently explore the search space. This will allow high possibility to obtain better solutions. We also believe that the results obtained can relate to the value of the conflict matrix density (see Table 1). The higher value of the conflict matrix density shows the higher the number of the exams that conflict with each other, thus less and sparsely distributed solution points are available in the solution space. This is indicated by the results obtained for the *ear-f-83* and *hec-s-92* datasets where the conflict matrix densities are higher compared to the other datasets used in this experiment. The results also show the strength of combining several neighbourhood structures where it helps to compensate against the ineffectiveness of using each type of neighbourhood structure in isolation and offer flexibility for the search algorithm to explore different regions of the solution space. This also has been aided by the implementation of a tabu list within a search process as a mechanism to escape from the cycle and to jump the barrier from one solution point to another in order to obtain a better solution. We also believed that if the algorithm is equipped with intensification and diversification strategies can help the algorithm to better explored the search space especially for the datasets with higher conflict density. However, in general, a tabu-based memetic algorithm is able to obtain some of best known results and comparable on the rest.

6. Conclusion

The overall goal of this paper is to investigate a tabu-based memetic approach for the examination timetabling problem. Preliminary comparisons indicate that our approach outperforms the current state of the art on four benchmark problems. The key feature of our approach is the combination of a multiple neighbourhood structures and a search approach that incorporates the concept of a tabu list. Our future work will be aimed to testing this algorithm on International Timetabling Competition datasets (ITC2007) introduced by University of Udine. We believe that the performance of a tabu-based memetic algorithm for the examination timetabling problem can be improved by applying advanced memetic operators and by intelligently selects the appropriate neighbourhood structures based on the current solution obtained.

References

1. S. Abdullah, S. Ahmadi, E.K. Burke and M. Dror, Investigating Ahuja-Orlin's large neighbourhood search approach for examination timetabling. OR Spectrum, 29(2), (2007) 351-372.
2. S. Abdullah and H. Turabieh. Generating university course timetable using genetic algorithm and local search. Proceeding of the 3rd International Conference on Hybrid Information Technology. (2008) 254-260.
3. S. Ahmadi, R. Barone, P. Cheng, P. Cowling and 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) 155-171.
4. H. Asmuni, E.K. Burke, J. Garibaldi and B. McCollum: Fuzzy multipleordering criteria for examination timetabling. In: E.K. Burke and M. Trick (eds). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3616. (2005) 334-353.
5. H. Asmuni, E.K. Burke, J. Garibaldi and B. McCollum:. A novel fuzzy approach to evaluate the quality of examination timetabling. In: E.K. Burke and H. Rudova (eds). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3867. (2007) 327-346.
6. B. Bilgin, E. Ozcan and E.E. Korkmaz. An experimental study on hyper-heuristics on exam timetabling. In: E.K. Burke and H. Rudova (eds). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3867. (2007) 394-412.
7. P. Boizumault, Y. Delon and L. Peridy. Constraint logic programming for examination timetabling. The Journal of Logic Programming, 26, (1996) 217-233.
8. E.K. Burke, J.P. Newall and R.F. Weare. A simple heuristically guided search for the timetable problem. The Proceedings of the International ICSC Symposium on Engineering of Intelligent System (EIS'98), Canada/Switzerland: ICSC/Academic Press, (1998) 574-579.
9. E.K. Burke and J.P. Newall. Solving examination timetabling problems through adaptation of heuristic orderings. Annals of Operations Research, 129, (2004) 107-134.

10. E.K. Burke, Y. Bykov, J.P. Newall and S. Petrovic. A time-predefined local search approach to exam timetabling problem. IIE Transactions, 36(6), (2004)509-528.
11. E.K. Buke and J.P Newall. Enhancing timetable solutions with local search methods. In E.K. Burke and P. De Causmaecker (eds). Selected Papers from 4th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 2740. (2003) 195-206.
12. E.K. Burke, J.P.Newall and R.F. Weare. A memetic algorithm for university exam timetabling. In E.K. Burke and P. Ross. (eds). Selected Papers from 1st International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 1153, (1996) 241-250.
13. E.K. Burke. J.P. Newall and R.F. Weare. Initialisation strategies and diversity in evolutionary timetabling. Evolutionary Computation, 6(1), (1998) 81-103.
14. E.K. Burke, A.J. Eckersley, B. McCollum, S. Petrovic and R. Qu. Hybrid variable neighbourhood approaches to university exam timetabling. Technical Report NOTTCS-TR-2006-2, School of Computer Science and Information Technology, University of Nottingham, United Kingdom. (2006).
15. E.K. Burke, M. Dror, S. Petrovic and R. Qu. Hybrid graph heuristics in hyper-heuristic applied to exam timetabling. In B.L. Golden, S. Raghavan and E.A. Wasil (eds). The Next Wave in Computing, Optimisation, and Decision Technologies. Springer. (2005) 79-91.
16. E.K. Burke, S. Petrovic and R. Qu. Case-based heuristic selection for timetabling problems. Journal of Scheduling, 9. (2006) 115-132.
17. E.K. Burke, B. McCollum, A. Meisels, S. Petrovic and R. Qu. A graph based hyper-heuristic for exam timetabling. European Journal of Operational Research, 176. (2007) 177-192.
18. E.K. Burke, G. Kendall and E. Soubeiga. A tabu-search hyper-heuristic for timetabling and rostering. Journal of Heuristics, 9(6), (2003) 451-470.
19. E.K. Burke and J.P. Newall. A multi-stage evolutionary algorithm for the timetable problem. IEEE Transactions on Evolutionary Computation, 3.1, (1999) 63-74.
20. E.K. Burke, D.G. Elliman, P.H. Ford and R.F. Weare. Examination timetabling in British universities - A survey. In E.K. Burke and P. Ross. (eds). Selected Papers from 1st International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 1153, (1996) 76-92.
21. M. Caramia, P. Dell'Olmo and G.F. Italiano. New algorithms for examination timetabling. Algorithms Engineering 4th International Workshop, Proceedings WAE 2001, Saarbrücken, Germany, Springer Lecture Notes in Computer Science, vol. 1982. (2001) 230-241.
22. M.W. Carter. A survey of practical applications of examination timetabling algorithms. Operations Research, 34(2): (1986) 193-202.
23. M.W. Carter and D.G. Johnson. Extended clique initialisatio in examintaion timetabling. Journal of Operational Research Society, 52: (2001) 538-544.
24. S. Casey and J. Thompson. GRASPing the examination scheduling problem. In E.K. Burke and P. De Causmaecker (eds). Selected Papers from 4th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 2740. (2003) 232-244.
25. D. Corne, P. Ross and H.L. Fang. Fast practical evolutionary timetabling. The Proceedings of Evolutionary Computing AISB Workshop, Springer Lecture Notes in Computer Science, vol. 865. (1994) 251-263.
26. P. Côté, T. Wong and R. Sabourin. A hybrid multi-objective evolutionary algorithm for the uncapacitated exam proximity problem. In: E.K. Burke and M. Trick (eds). Selected Papers from the 5th International Conference on the Practice and Theory of

Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3616. (2005) 294-312.

27. P. David. A constraint-based approach for examination timetabling using local repair technique. In E.K. Burke and M.W. Carter (eds). Selected Papers from the 2nd International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 1408. (1998) 169-186.
28. L. Di Gaspero. Recolor, shake and kick: A recipe for the examination timetabling problem. The Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling (PATAT IV), Gent, Belgium, (2002) 404-407.
29. L. Di Gaspero L and A. Schaerf. Tabu search techniques for examination timetabling. In E.K. Burke and W. Erben. (eds). Selecter Papers from the 3rd International Conference on the Practice and Theory of Automated Timetabling Springer Lecture Notes in Computer Science, vol. 2079, (2001) 104-117.
30. K.A. Dowsland and J.M. Thompson. Ant colony optimisation for the examination scheduling problem. Journal of the Operational Research Society, 56, (2005) 426-438.
31. M. Eley. Ant algorithms for the exam timetabling. In: E.K. Burke and H. Rudova (eds). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3867. (2007) 364-382.
32. W. Erben. A grouping genetic algorithm for graph colouring and exam timetabling. In E.K. Burke and P. De Causmaecker (eds). Selected Papers from 4th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 2740. (2001) 132-156.
33. E. Ersoy, E. Ozcan and A.S. Etaner. Memetic algorithms and hyperhill-climbers. In Proceedings of the 3rd Multidisciplinary International Conference on Scheduling: Theory and Applications. (2007) 159-166.
34. P. Hansen and N. Mladenović. Variable neighbourhood search. European Journal of Operational Research, 130, (2001) 449-467.
35. G. Kendall and N.M. Hussin. An investigation of a tabu search based hyper-heuristic for examination timetabling. In G. Kendall, E.K. Burke and S. Petrovic (eds). Selected Papers from Multidisciplinary Scheduling: Theory and Applications. (2005) 309-328.
36. R. Lewis. A survey of metaheuristic-based techniques for university timetabling problems. OR Spectrum, 30(1). (2008) 167-190.
37. S.L.M. Lim. A broker algorithm for timetabling problem. In E.K. Burke and P. De. Causmaecker. (eds). Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling. (2002) 372-386.
38. H.R. Lourenco, O. Martin and T. Stutzle. Iterated local search. In F. Glover and G.A. Kochenberher (eds). Handbook of Metaheuristics. Boston: Kluwer Academic. (2003) 321-354.
39. M.R. Malim, A.T. Khader and A. Mustafa. Artificial immune algorithms for university timetabling. In: E.K. Burke and H. Rudova (eds). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3867. (2006) 234-245.
40. B. McCollum. A perspective on bridging the gap between theory and practice in university timetabling. In: E.K. Burke and H. Rudova (eds). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3867. (2007) 3-23.
41. B. McCollum, E.K. Burke, P. McMullan. A review and description of datasets, formulations and solutions to the University Course Timetabling Problem. Journal of Scheduling. (2009, submitted)

42. L.T.G. Merlot, N. Boland, B.D. Hughes and P.J. Stuckey. A hybrid algorithm for the examination timetabling problem. In E.K. Burke and P. De Causmaecker (eds). Selected Papers from 4th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 2740. (2003) 207-231
43. N. Mladenović and P. Hansen. Variable neighbourhood search. Computers and Operations Research, 24(11): (1997) 1097-1100.
44. Z. Naji Azimi. Comparison of metaheuristic algorithms for examination timetabling problem. Applied Mathematics and Computation, 16(1-2). (2004) 337-354.
45. L. Pacquete and T. Stützle. Empirical analysis of tabu search for the lexicographic optimisation of the examination timetabling problem. The Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling (PATAT IV), Gent, Belgium, (2002) 413-420.
46. S. Petrovic and E.K. Burke. Chapter 45: University timetabling. In: J. Leung (ed): Handbook of Scheduling: Algorithms, Models, and Performance Analysis,CRC Press, April (2004).
47. R. Qu and E.K. Burke. Adaptive decomposition and construction for for university timetabling problems. Jn Multidisciplinary International Scheduling: Theory and Applications (MISTA'07). (2007) 418-425.
48. R. Qu and E.K. Burke. Hybridisations within a graph based hyper-heuristic framework for university timetabling problems. Journal of Operational Research Society. (2008, accepted)
49. R. Qu, E.K. Burke, B. McCollum and L.T.G. Merlot. A survey of search methodologies and automated system development for examination timetabling. Journal of Scheduling, 12. (2009) 55-89.
50. P. Ross, D. Corne and H.L. Fang. Improving evolutionary timetabling with delta evolution and directed mutation. The Proceedings of the Conference of Parallel Problem Solving in Nature III, Springer Lecture Notes in Computer Science, vol. 866. (1994) 556-565.
51. P. Ross, E. Hart and D.Corne. Some observations about GA based timetabling. In E.K. Burke and M.W. Carter (eds). Selected Papers from the 2nd International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 1408. (1998) 115-129.
52. P. Ross, J.G. Marin-Blazquez and E. Hart. Hyper-heuristics applied to class and exam timetabling problems. In Proceedings of the 2004 Congres on Evolutionary Computation (CEC 2004), (2004) 1691-1698.
53. H. Terashima-Marín, P. Ross and M. Valenzuela-Rendón. Clique-based crossover for solving the timetabling problem with GAs. The Proceeding of the Congress on Evolutionary Computation. Washington, D.C., (1999) 1200-1206.
54. J.M. Thompson and K.A. Dowsland. Variants of simulated annealing for the examination timetabling problem. Annals of Operations Research, 63, (1996) 105-128.
55. O. Ulker, E. Ozcan and E.E. Korkmaz. Linear linkage encoding in grouping problems: applications on graph colouring and timetabling. In: E.K. Burke and H. Rudova (eds). Selected Papers from the 5th International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 3867. (2007) 347-363.
56. G.M White and B.S Xie. Examination timetables and tabu search with longer-term memory. In E.K. Burke and W. Erben. (eds). Selected Papers from 3rd International Conference on the Practice and Theory of Automated Timetabling. Springer Lecture Notes in Computer Science, vol. 2079. (2001) 85-103.

57. B. McCollum, A. Schaerf, B. Paechter, P. McMullan, R. Lewis, A. Parkes, L. Di Gaspero, R. Qu, E.K. Burke, E.K. Setting the research agenda in automated timetabling: The second international timetabling competition, Accepted for publication to INFORMS Journal of Computing (2007).
58. W.E. Hart, N. Krasnogor, J.E. Smith, “Memetic Evolutionary Algorithms”. *Recent Advances in Memetic Algorithms: Studies in Fuzziness and Soft Computing*. Springer-Verlag, (2004) 3-27.
59. C.Y. Cheong, K.C. Tan, B. Veeravalli, Solving the exam timetabling problem via a multi-objective evolutionary algorithm: A more general approach. In IEEE symposium on computational intelligence in scheduling, Honolulu, HI, USA. (2007) 165–172.