A Tabu based Large Neighbourhood Search Methodology for the Capacitated Examination Timetabling Problem

Salwani Abdullah[1], Samad Ahmadi[2], Edmund K. Burke[1], Moshe Dror[3] and Barry McCollum[4]

[1] Automated Scheduling, Optimisation and Planning Research Group, School of Computer Science & Information Technology, University of Nottingham, Jubilee Campus, Wollaton Road, Nottingham NG8 1BB, United Kingdom.

[2]School of Computing, De Montfort University, The Gateway, Leicester LE1 9BH, United Kingdom.

[3]MIS Department, College of Business and Public Administration, University of Arizona, Tucson, Arizona 85721, USA

[4]Department of Computer Science, Queen's University Belfast, Belfast BT7 1NN, United Kingdom

{sqa@cs.nott.ac.uk, sahmadi@dmu.ac.uk, ekb@cs.nott.ac.uk, mdror@bpa.arizona.edu, b.mccollum@qub.ac.uk}

Neighbourhood search algorithms are often the most effective known approaches for solving partitioning problems. In this paper we consider the capacitated examination timetabling problem as a partitioning problem and present an examination timetabling methodology which is based upon the large neighbourhood search algorithm that was originally developed by Ahuja and Orlin. It is based on searching a very large neighbourhood of solutions using graph theoretical algorithms implemented on a so called improvement graph. In this paper, we present a tabu based large neighbourhood search, in which the improvement moves are kept in a tabu list for a certain number of iterations. We have drawn upon Ahuja-Orlin's methodology incorporated with tabu lists and developed an effective examination timetabling solution scheme which we evaluated on capacitated problem benchmark data sets from the literature. The capacitated problem includes the consideration of room capacities and, as such, represents an issue that is of particular importance in real world situations. We compare our approach against other methodologies that have appeared in the literature over recent years. Our computational experiments indicate that the approach we describe produces the best known results on a number of these benchmark problems.

Keywords: examination timetabling, large neighbourhood, improvement graph, tabu search.

Correspondence to: Salwani Abdullah

Introduction

This paper investigates a large neighbourhood solution approach based on tabu search and its application to capacitated examination timetabling. The objective is to see how the hybridisation of a large neighbourhood algorithm, a tabu list and adaptive memory can enhance the solution quality in capacitated examination timetabling.

The idea of tabu search was proposed by Fred Glover (1987). Glover and Laguna define tabu search as:

"*a metaheuristic that guides a local heuristic search procedure to explore the solution space beyond local optimality*"

The basic mechanism of a tabu search is to iteratively explore a subset of the neighbourhood of the current solution. The member of the neighbourhood that gives the minimum value the of cost function (assuming minimisation) becomes the new solution. However, to prevent the search from getting stuck in local optima, a so-called tabu-list is maintained i.e. a list that contains moves that satisfy some tabu restriction criterion. These previously visited moves are forbidden to be performed for a certain number of iterations (called the tabu tenure). The tabu tenure determines how long a move remains tabu-active. However, a mechanism called the aspiration-criterion is sometimes used to override the tabu status of a move. A common aspiration criterion is a better improvement of the cost function i.e. a tabu move is changed to a non-tabu move if it produces a better solution. A basic introduction to tabu search can be found in Gendreau and Potvin (2005) and a comprehensive treatment can be found in Glover and Laguna (1997).

Schaerf (1999[a]) applies tabu search techniques in scheduling lectures to timeslots for a large high-school. A variable size of tabu list is used. Each move is added into the tabu list where the size of the tabu list is randomly selected from a pre-determined range. Therefore the tabu tenure varies for each move in the tabu list. A common aspiration criterion (as mentioned in the above paragraph) is employed. Experimental results show that the algorithm is able to produce a timetable that is able to schedule 90-95% of the lectures (and is better than the manual timetable). Di Gaspero and Schaerf (2001) present an examination timetabling algorithm that is based on tabu search and graph colouring heuristics. In order to guide the search to explore different areas of the solution space, they modified the objective function by changing the weights. A dynamic size of the tabu list in the interval $k_{min} - k_{max}$ is used to store the most recently accepted moves. The same aspiration criterion as in Schaerf (1999[a]) is employed and the algorithm is tested on standard benchmark and random instances on examination timetabling.

White and Xie (2001) implemented a tabu search algorithm which is called OTTABU. It used both recency-based short-term memory and frequency-based longer-term memory to improve the solution quality. Burke *et al* (2003[a]) applied a tabu search hyper-heuristic technique for nurse rostering and course timetabling problems (see Burke *et al*, 2003[b] for an overview of hyperheuristic approaches). In this algorithm, a set of low level heuristics are competing with each other. When a heuristic has been applied, the change in the cost function value from the previous to a new solution is noted. A variable length dynamic tabu list of low level heuristics is maintained which stops certain heuristics from being employed for a certain duration. The status of the heuristics in the tabu list will be changed from tabu active to non-tabu active if there is an improvement in the cost function. The authors believe that there is no point in keeping a heuristic tabu once the current solution has been updated. Experimental results show that this technique is capable of producing acceptable solution qualities across both nurse rostering and course timetabling problems.

Kendall and Hussin (2004a) employed a tabu search based hyper-heuristic for examination timetabling. Their algorithm uses a fixed length of tabu list which is equal to the number of low-level heuristics. The heuristics that have been applied become tabu. Instead of adding moves in the tabu list, this algorithm stores information about each heuristic (i.e. heuristic number, recent change in cost function, CPU time taken to run the heuristic and how long the heuristic should remain tabu). Preliminary results reported that the method was unable to beat the best known results from the literature on standard benchmark problems but the algorithm can produce good quality solutions across a variety of problem instances. The authors continued this research (Kendall and Hussin, 2004b) by exploring three different types of tabu-based hyper-heuristic (i.e. a simple tabu search hyper-heuristic that considers all non-tabu heuristics and applies the heuristic that has the best improvement only in the cost function, a tabu search hyper-heuristic with hill climbing and a tabu search hyper-heuristic with great deluge). They applied these methods to a very large problem from the MARA University of Technology and standard benchmark examination timetabling datasets. Experimental results show that the algorithm can produce at least an 80% improvement compared to the manual solution for the MARA dataset and it can also improve the results obtained from Kendall and Hussin (2004a). Other examples of papers which discuss tabu search in timetabling are Dowsland (1998), White *et al* (2004), White and Xie (2001) and Wright (2001).

In our paper, we use a fixed length of tabu list and we make tabu any examinations that have been involved in a move. A detailed implementation of our algorithm is presented in the search algorithm section.

When we use the term *large neighbourhood structure* we refer to the size of the neighbourhood which is "*very large*" with respect to the size of the input data (see Ahuja *et al*, 2002). In this paper, we discuss a very large-scale neighbourhood partitioning problem approach for timetabling. The most popular neighbourhood for the partitioning problem is a two-exchange neighbourhood (see Ahuja *et al*, 2000). The cyclic-exchange neighbourhood which was proposed by Thompson and Orlin in 1989 is a generalization of the two-exchange neighbourhood. It has substantially more neighbours compared to the two-exchange neighbourhood. Generally, the quality of locally optimal solutions obtained from the larger neighbourhood structure is better than those obtained from a smaller neighbourhood structure (see Ahuja *et al*, 2000). However, the price to pay is the amount of computational time that is required. We shall see that this plays a role in capacitated examination timetabling. In order to balance the quality of solution and the computational time required and to demonstrate that the larger neighbourhood search produces a better quality of (local optimal) solution, Ahuja *et al* (2000) applied a network flow optimization methodology on the capacitated minimum cost spanning tree problem. Ahuja *et al* (2001) used a multiexchange neighbourhood structure for the same problem, and in 2003 (Ahuja *et al*, 2003) they explored a composite neighbourhood structure (a unification of a node-based and tree-based exchange), in which they found that the result obtained from the composite neighbourhood is better than a node-based neighbourhood structure alone. The interested reader can find more details on large neighbourhood methods in Ahuja *et al* (2000, 2001, 2002, 2003).

In this paper, we will concentrate on capacitated examination timetabling problems. A survey of examination timetabling in British universities conducted by Burke *et al* (1996^b) concluded that examination timetabling software needs to cater for different demands that vary significantly from one institution to another. Carter and Laporte (1996) defined the examination timetabling problems as:

"*the assigning of examinations to a limited number of available time periods in such a way that there are no conflicts or clashes*".

The above term "*no conflict or clashes*" from Carter and Laporte (1996) is an example of a *hard* constraint (i.e. one which should not be violated). A solution (timetable) that obeys all the hard constraints is often called a feasible solution. In this paper, we consider the following hard constraints: (1) no student can sit in two examinations simultaneously; and (2) the schedule examinations must not exceed the classroom capacity. There is no attempt to divide students between different classrooms that might be available for the examination. Soft constraints are those which are considered to be desirable but which can be violated if necessary. A particularly common soft constraint is the goal of spreading examinations as evenly as possible over the time schedule (see Burke *et al*, 1996^a). The quality of the feasible solution can be measured by the degree to which these soft constraints are satisfied. Indeed, this is exactly the measure that we use in the experiments carried out in this paper.

A comprehensive overview of the wide variety of approach and techniques that have been applied to examination timetabling over the years can be found by consulting de Werra (1985), Carter (1986), Bardadym (1996), Burke *et al* (1996^b), Carter and Laporte (1996), Burke *et al* (1997), Schaerf (1999^b), Burke and Petrovic (2002) and Petrovic and Burke (2004).

In this paper, we treat the examination timetabling problem as a variant of the partitioning problem. We incorporate a tabu list with our large neighbourhood examination timetabling algorithm (as presented in Abdullah *et al*, 2006) for capacitated examination timetabling problems. This paper is organised as follows: The next section formally presents the description of the examination timetabling problem. The model adapted from Ahuja *et al* (2001) is outlined in the modelling section. The solution search strategy that we employ is discussed in the search algorithm section. Our results are presented and evaluated in the experimental results and analysis section followed by some concluding remarks in the conclusion and future work section.

Problem Description

The description of the examination timetabling problem that we present here is adapted from the one given in Burke *et al* (2004). The input can be outlined as follows:

- N is the number of examinations
- E_i is the examinations, $i \in \{1,\ldots,N\}$
- B is the set of all N examinations, $B = \{E_1,\ldots,E_N\}$
- T is the given number of available timeslots
- t_k is the timeslot for examination k. The timeslots are denoted by positive integers.
- $C = (c_{ij})_{NxN}$ is a *conflict* matrix where each element denoted by c_{ij}, $i,j \in \{1,\ldots,N\}$ is the number of students taking both examinations i and j.

We will be considering the capacitated version of the problem. As mentioned above, we are concerned with the following two hard constraints: (i) All assignments should be clash-free and (ii) the maximum capacity of a room cannot be violated. We assume that a weekly programme of examinations starts on Monday and is finished on Saturday with no examination on Sunday. A complete examination timetable will require a number of weeks (often 3 or 4). Each week day has 3 timeslots, Saturday has 1 timeslot and Sunday has none. We use the day vector representation that was employed in Abdullah *et al* (2006) as:

(1,1,1,2,2,2,3,3,3,4,4,4,5,5,5,6,8,8,8,9,9,9,10,10,10,11,11,11,12,12,12,13,15,15,15,…)

The corresponding timeslot vector is $(1,2,3,4,5,6,7,8,9,\ldots,T)$. Note that the one "6" entry and the one "13" entry (see Figure 1) corresponds to the timeslot on the first Saturday and the timeslot on the second Saturday respectively. The entries corresponding to Sundays are missing altogether. The day for a particular timeslot t (where $t \in \{1,\ldots,T\}$) is represented by d_t. For example, d_4 would be day 2, d_7 would be day 3 and d_{17} would be day 8.

We use the objective function as described in Burke *et al* (1996[a]) which minimizes the number of students having two examinations in a row on the same day. The symbolic representation of the relationship in our model is stated below.

The objective function is to minimise:

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

where

$$Adj(t_i, t_j) = \begin{cases} 1 & if\,(|\,t_i - t_j| = 1) \wedge (d_{t_i} = d_{t_j}) \\ 0 & otherwise \end{cases} \tag{2}$$

subject to

$$\sum_{i-1}^{N-1} \sum_{j=i+1}^{N} c_{ij} \cdot x(t_i, t_j) = 0 \text{ where } x(t_i, t_j) = \begin{cases} 1 & if \quad t_i = t_j \\ 0 & otherwise \end{cases} \tag{3}$$

Equation (1) represents our objective function to minimize the number of students having examinations in adjacent periods on the same day. Equation (2) represents a function which tells us if timeslots on the same day are adjacent and Equation (3) represents a clash-free requirement so that no student is asked to sit two examinations at the same time. The objective function is also subject to

$$Student_t \leq Seats \text{ for } t = 1,\ldots,T \tag{4}$$

where $Student_t$ is the number of students taking examinations in timeslot t and *Seats* is the number of seats available in each timeslot. The inequality in (4) represents the room capacity for the total number of students in each timeslot. The total number of students taking examinations in each timeslot $Student_t$ should be less than or equal to *Seats*.

Modelling

This section discusses the solution process we have developed for the capacitated examination timetabling problem. The process runs through several stages including the representation of the problem where we treat the problem as a variant of a partitioning problem. Secondly, we define our large neighbourhood structure through a cyclic exchange operation. Next, we proceed to a construction of the improvement graph that is employed within the large neighbourhood search algorithm and lastly we describe how to determine the profitable moves in the improvement graph using a network flow optimisation technique. The details of each stage can be found in the following:

Partitioning Problem

Let $\{S_t: t \in \{1,\ldots,T\}, S_t \subseteq B\}$ denote a feasible partition of examinations that are to be scheduled at time t. The partition $\{S_t\}$ divides the set B so that $\cup_{t\in\{1,\ldots,T\}} S_t = B$ and $S_q \cap S_t = \varnothing$ for all q, $t \in \{1,\ldots,T\}$, $q \neq t$. The partition subsets will be referred to as *cells*.

Creating Neighbours

We create a similar neighbourhood structure to that discussed in Abdullah *et al.* (2006). The process moves a single examination between a number q, of cells. Note that $1 \le q \le T$, where T is a given maximum on the number of cells (which is equal to the maximum number of timeslots). Such a move is known as a cyclic exchange neighbourhood. We use the following notation to denote this : $i_r \to i_s \to i_t \to \ldots \to i_q \to i_r$, where i_r, i_s, i_t ,…, i_q are examinations which belong to different cells and r,s,t,q are all $\le T$ and are all distinct. The notation tells us that examination i_r moves from its cell to the cell containing i_s and so on until i_q replaces i_r. We can define path exchanges in a similar manner but without returning to the starting cell. Both exchanges lead to the generation of a very large neighbourhood. Abdullah *et al* (2006) present more details and a simple illustrative example of the procedure.

Improvement Graph

The improvement graph, G was introduced in Thompson and Orlin (1989) and explored in Thompson and Psaraftis (1993). An improvement graph is constructed by taking every pair of examinations i_1 and i_2 from different cells as the nodes of the graph. Directed edges between i_1 and i_2 exist if and only if the new cell generated by $i_1 \to i_2$ is feasible. A cell is called feasible if the hard constraints are not violated. The cost of a directed arc between i_1 and i_2 is defined as:

(cost of ((“*examination* i_1” ∪ “*cell to which examination* i_2 *belongs*”) \ “*examination* i_2”)) – (cost of (“*cell to which examination* i_2 *belongs*”))

The *cost of* “*examination* i_1” is calculated using Expression (1). It represents the number of students who are forced to take a conflicting examination in a period adjacent to “*examination* i_1”. The cost of “*examination* i_2” is calculated in the same manner. The cost of the “*cell to which examination* i_2 *belongs*” is a summation of the cost of all examinations in the cell. Note that the term “*inserted examinations*” is used in the search algortihm section to discuss a number of examinations rather than the single case (“*examination* i_1”) here. Abdullah *et al* (2006) present a simple illustrative example of an improvement graph in examination timetabling.

Identifying the Negative Cost Partition-disjoint Cycle

A valid cycle (or valid path) in the improvement graph is referred to as a negative cost partition-disjoint cycle. It represents an improving move (or moves). In the context of examination timetabling, these improving moves represent (improving) exchanges of examinations (either a path or cycle exchange) from one timeslot to another. To determine these exchanges, we draw upon a network flow optimization technique (as discussed in Abdullah *et al*, 2006) by heuristically determining negative cost partition-disjoint cycles for the improvement graph using a modified shortest path label-correcting algorithm (Ahuja *et al*, 1993).

The Search Algorithm

The pseudo code for our approach is illustrated in Figure 1 which is started with a feasible solution *Sol*. Cells of examinations are produced based on timeslots (as mentioned earlier) followed by the construction of the improvement graph where it is constructed once. In the repetition loop, the negative cost partition-disjoint cycle for the improvement graph is obtained by executing the modified shortest path label-correcting algorithm (Ahuja *et al*, 1993). It is run several times with a different *source examination* in the improvement graph, since the success of finding a valid cycle is related to the *source examination* from which the search is initiated. The *source examinations* used are chosen from the "*inserted examinations*" in the improvement graph because they have an *out going* directed arc. The *out going* directed arc is important to direct further search in finding a valid cycle. We re-evaluate the quality of the new solution $f(Sol^*)$ once the negative cost partition-disjoint cycle is obtained. The new solution is compared to the best solution and it is accepted if there is an improvement (including a zero improvement). The best solution Sol_{best} is then replaced with Sol^*. The reason we accept the zero improvement is because the new solution might be different from the best solution even though the cost function is producing the same result (i.e. the improvement is zero). The algorithm will always accept an improved solution and a worse solution is accepted with a certain probability with the aim to escape from local optima by using an exponential Monte Carlo acceptance criterion which has been shown to work well in the application of component placement sequencing for a multi head placement machine (see Ayob and Kendall, 2003). Exponential Monte Carlo is similar to the acceptance criterion in a simulated annealing approach. The difference is that no cooling schedule is required. It is only based on the solution quality where the new solution Sol^* is accepted if the generated random number, *RandNum*, in [0,1] is less than the probability $e^{-\delta}$ where δ is the difference between the old and new solutions (i.e. $\delta = f(Sol^*) - f(Sol)$). It will exponentially increase the acceptance probability if δ is small. If a worse solution is not accepted by the exponential Monte Carlo acceptance criterion for a certain number of iteration (which is equal to the *not_improving_constant*), then the algorithm will terminate. The details of the search algorithm are also discussed in more depth in Abdullah *et al* (2006).

PUT FIGURE 1 HERE

Our tabu search algorithm uses only short term memory. We add any examinations that have been involved in a move to the tabu list, denoted as $TabuList_i$ $(i \in \{1,\ldots,N\})$. These examinations are not allowed to be part of any exchange for a certain number of iterations (the tabu tenure) so that we can look at the possibility of other examinations to be considered in performing a cyclic exchange and constructing an improvement graph that may result in better profitable exchanges with respect to the objective function. The tabu tenure is decreased after each iteration until it reaches zero. All tabu examinations will change to non tabu status when the tabu tenure is zero. In these experiments, we set the tabu tenure to be 2, 4 and 6. The determination of these values was based upon a series of experiments.

The improvement graph is updated after the cyclic (or path) exchange is performed. The cells which play a part in the process (referred to as the *AffectedCells*) can be determined by consulting the cyclic (or path) exchanges. So, the costs for all the directed arcs that are not connected to the *AffectedCells* remain unchanged. We refer to these arcs as *OriginalArcs*. This enables us to save the time taken to create and calculate the cost for *OriginalArcs*. We then proceed by creating the new directed arcs referred to as *TNewDirArcs1* in case 1 (see Figure 2) by inserting an examination (from the *AffectedCells*) to another cell and ejecting one examination (the ejected examination should be different from the examinations in the *TabuList*) and by calculating the cost for the new directed arcs. The other new directed arcs in case 2 are referred to as *TNewDirArcs2* (see Figure 2) and are created like the *TNewDirArcs1* in case 1, but in the reverse way i.e. we insert an examination (which is not in the *TabuList)* from one cell to the *AffectedCells* and, at the same time, eject another examination from this *AffectedCells* and this ejected examination should not be in the *TabuList*. Then we calculate the cost for the *TNewDirArcs2*. The combination of the *OriginalArcs*, *TNewDirArcs1* and *TNewDirArc2* will form a new improvement graph *G*. Figure 2 shows the pseudo code for updating the improvement graph in our method.

PUT FIGURE 2 HERE

Experimental Results and Analysis

The algorithm was tested on standard benchmark problems using the Carter *et al.* (1996) data collection and the University of Nottingham dataset which was introduced by Burke *et al* (1996[a]) and which also includes the room capacities requirement.

The hybridised tabu search large neighbourhood method described in this paper was run on six datasets. For each dataset we experimented with tabu tenure 2, 4 and 6. The characteristics of the datasets used (taken from Carter *et al*, 1996 and Burke *et al*, 1996[a]) are presented in Table 1. The number of examinations which conflict with other examinations is summed up and divided by the total number of examinations, to give an average conflict matrix density for each dataset in Carter *et al* (1996). We used the same formula as above to calculate the conflict density matrix for the *nott-94* dataset (since it is not available in Burke *et al*, 1996[a]).

PUT TABLE 1 HERE

Our approach was implemented in Visual C++. We ran the experiment overnight (which takes approximately twelve hours) for each of the datasets on an Athlon machine with a 1.2 GHz processor and 256 MB RAM. Note that running a system overnight is not unreasonable in the context of examination timetabling where the timetables are usually produced months before the actual schedule is required. Table 2 shows the computational results of our algorithm compared to other published results for these benchmark datasets. The best results are shown in bold. We compare our results with results from:

- Burke *et al* (1996[a]) which employed a memetic algorithm.
- Caramia *et al* (2001) which implemented a greedy constructive heuristic with an optimisation step.
- Di Gaspero and Schaerf (2001) which employed tabu search.
- Merlot *et al* (2003) which applied a hybridization of constraint programming, simulated annealing and hill climbing.
- Abdullah *et al* (2006) which employed large neighbourhood with a network flow optimization technique.

PUT TABLE 2 HERE

Our objective here is to demonstrate that the tabu based large neighbourhood search methodology is able to produce good results for capacitated examination timetabling problems. The method produces the best known result in the literature on two of the six problems (tieing with Merlot *et al*, 2003 on *tre-s-92*). Our results are better than Burke *et al.* (1996[a]) on all of the datasets. We have better results than Caramia *et al* (2001) in five of six datasets. Our results are better than Di Gaspero and Schaerf (2000) in three of the six datasets. However, we are better than Merlot *et al* (2003) in just one instance with a tie on the *tre-s-92* dataset.

We are particularly interested to compare our results with the results in Abdullah *et al.* (2006). In Abdullah *et al* (2006) we did not employ a tabu list. In that case, we can consider the tabu tenure to be zero. Since the tabu search based method is superior in four out of five problems there is evidence to suggest that we could use a tabu list to guide the search process. Note that Abdullah *et al* (2006) do not attempt the *nott-94* dataset. In our tabu based large neighbourhood search, we apply a tabu restriction where examinations will be tabu active if examinations in a cyclic (or path) exchange have been accepted to update the current solution. The examination(s) will remain tabu active for a number of iterations which is equal to the tabu tenure. We made the examinations in a cyclic (or path) exchange tabu because we want to direct the search to other parts of the search space in the improvement graph. The best results for four of the datasets (we do not consider the *nott-94* dataset in this comparison since Abdullah *et al*, 2006 do not attempt this dataset) is when the value of TT (representing the tabu tenure) is 2 (ties with TT is 4 on *tre-s-92* and *uta-s-92* datasets) and for the other dataset it is when TT is zero (from Abdullah *et al*, 2006). It is interesting to note that for the *kfu-s-93* dataset, we obtain the best result when the value of TT is zero and that this problem instance has the lowest conflict matrix density (0.06). The lower conflict matrix density signifies that fewer examinations are conflicting with each other. This implies that we might have more feasible solution points in our search space. So, we do not need a tabu list to direct the search to other parts of the search space in the improvement graph. The higher the value of TT, the longer the examinations will remain in the tabulist. This limits the search space. We notice that a higher value of TT makes the solution considerably worse and thus more difficult to improve. Figures 3 and 4 show the performance of our algorithm with a different value of TT on two datasets (*tre-s-92* and *kfu-s-93*).

PUT FIGURES 3 AND 4 HERE

This graph demonstrates how our algorithm explores the search space in the improvement graph. The x-axis represents the number of iterations while the y-axis represents the average penalty cost per student. The curves move up and down because at every iteration we accept a best solution (or worse solution with some probability). In Abdullah *et al* (2006), the solutions are trapped in a cycle. The use of the tabu list in this algorithm helps to avoid cycling during the search process. To show the effect of using the tabu based large neighbourhood search in avoiding cycling in the search process, we present the graph (see Figure 5) of the solution for the *car-f-92* dataset and the solution taken from Abdullah *et al.* (2006) in which we consider the value of TT to be zero. The tabu based large neighbourhood search can find the solutions (in most cases) with better local optima.

PUT FIGURE 5 HERE

Conclusions and Future Work

In this paper, we modelled the capacitated examination timetabling problem as a variant of the partitioning problem consisting of cells of examinations that are scheduled in a number of timeslots. We created a neighbour through a cyclic exchange operation, in which the created neighbourhood structure is extremely large. We presented the hybrid tabu based large neighbourhood search in order to direct the search to other parts of the improvement graph and to avoid cycling during the search process. We identified negative cost partition-disjoint cycles in the improvement graph using a modified shortest path label-correcting algorithm. The experiments carried out showed that our algorithm performed competitively in comparison to the best published results in the literature and to our previous results in Abdullah *et al* (2006). Indeed, it is able to provide the best known results on two of the six standard benchmarks problems in this area. It shows that the hybridization of the meta-heuristic and network flow optimization technique is a very promising technique for tackling the capacitated examination timetabling problem. Since the neighbourhood structure generated through the cyclic exchanges operation highlighted in this paper increases exponentially with the size of the input, we found that the drawback of our algorithm is an expensive computational time in finding a valid cycle (or path) in the improvement graph. Although computational time is not a critical feature of examination timetabling, future work will aim to shorten the computational time. We also want to rigorously analyse the performance of this approach by varying the tabu tenure during the search process in order to provide a balance between intensification and diversification of the search space in the improvement graph or to apply a tabu relaxation. This would entail reinitialising the tabu list after a number of non-improving iterations and starting the search from an elite solution.

Acknowledgement

This work has been supported by the Public Services Department of Malaysia (JPA) and the University Kebangsaan Malaysia (UKM). Prof. Dror's contribution was funded by an EPSRC Visiting Fellowship (GR/S071241/01).

References

Abdullah S., Ahmadi S, Burke EK and Dror M (2006). Investigating Ahuja-Orlin's Large Neighbourhood Search Approach for Examination Timetabling. Accepted for publication in *OR Spectrum*, to appear 2006.

Ahuja RK, Magnanti TL and Orlin JB (1993). *Network Flows: Theory, Algorithms, and Applications*. ISBN 1000499012. Prentice Hall: New Jersey.

Ahuja RK, Orlin JB and Sharma D (2000). Very Large Scale Neighbourhood Search. *International Transaction in Operational Research* **7**: 302-317.

Ahuja RK, Orlin JB and Sharma D (2001). Multiexchange Neighbourhood Search Algorithm for Capacitated Minimum Spanning Tree Problem. *Mathematical Programming* **91**: 71-97.

Ahuja RK, Ergun 0, Orlin JB and Punnen AO (2002). A Survey of Very Large-scale Neighbourhood Search Techniques. *Discrete Applied Mathematics* **123**: 75-102.

Ahuja RK, Orlin JB and Sharma D (2003). A Composite Neighbourhood Search Algorithm for Capacitated Spanning Tree Problem. *Operations Research Letters* **31**: 185-194.

Ayob M and Kendall G (2003). A Monte Carlo Hyper-Heuristic To Optimise Component Placement Sequencing For Multi Head Placement Machine. In: *Proceedings of the International Conference on Intelligent Technologies*, InTech'03, Chiang Mai, Thailand, pp 132-141.

Bardadym VA (1996). Computer-Aided School and University Timetabling: The New Wave. In Edmund Burke and Peter Ross, editors, *The Practice and Theory of Automated Timetabling*. Lecture Notes in Computer Science 1153. Springer-Verlag: Berlin, pp 22-45.

Burke EK and Petrovic S (2002). Recent Research Directions in Automated Timetabling. *Eur J Opl Res* **140**: 266-280.

Burke EK, Newall JP and Weare RF (1996^{a}). A Memetic Algorithm for University Exam Timetabling. In Edmund Burke and Peter Ross, editors, *The Practice and Theory of Automated Timetabling*. Lecture Notes in Computer Science 1153. Springer-Verlag: Berlin, pp 3-21.

Burke EK, Elliman D, Ford P and Weare RF (1996^{b}). Examination Timetabling in British Universities – A Survey. In Edmund Burke and Peter Ross, editors, *The Practice and Theory of Automated Timetabling*. Lecture Notes in Computer Science 1153. Springer-Verlag: Berlin, pp 76-92.

Burke, EK, Jackson KS, Kingston JH and Weare RF (1997). Automated Timetabling: The State of the Art. *The Computer Journal* **40**(9): 565-571.

Burke, E.K., Kendall. G., Soubeiga. E. (2003^{a}) A Tabu-Search Hyperheuristic for Timetabling and Rostering. *Journal of Heuristics* **9**(6): 451-470.

Burke, E., G. Kendall, J. Newall, E. Hart, P. Ross, and S. Schulenburg. (2003^{b}). *Handbook of Metaheuristics*, chapter 16, Hyper-Heuristics: An Emerging Direction in Modern Search Technology, Kluwer Academic Publishers, pp 457–474.

Burke EK, Bykov Y, Newall JP and Petrovic S (2004). A Time-Predefined Local Search Approach to Exam Timetabling Problem. *IIE Transactions on Operations Engineering* **36**(6), pp 509-528.

Caramia M., Dell'Olmo P and Italiano GF (2001). New Algorithms for Examination Timetabling. In Naher, S., Wagner, D., editors, *Algorithm Engineering 4th Int. Workshop, Proc. WAE 2000* Saarbrucken, Germany. Lecture Notes in Computer Science 1982. Springer-Verlag: Berlin, pp 230-241.

Carter MW (1986). A Survey of Practical Applications of Examination Timetabling. *Opns Res* **34**: 193-202.

Carter MW and Laporte G (1996). Recent Developments in Practical Examination Timetabling. In Edmund Burke and Peter Ross, editors, *The Practice and Theory of Automated Timetabling*. Lecture Notes in Computer Science 1153. Springer-Verlag: Berlin, pp 3-21.

Carter MW, Laporte G and Lee SY (1996). Examination Timetabling: Algorithmic Strategies and Applications. *J Opl Res Soc* **74:** 373-383.

de Werra D (1985). An Introduction To Timetabling. *Eur J Opl Res* **19**: 151-162.

Di Gaspero L and Schaerf A (2001). Tabu Search Techniques for Examination Timetabling. In Edmund Burke and Wilhelm Erbens, editors, *The Practice and Theory of Automated Timetabling*. Lecture Notes in Computer Science 2079. Springer-Verlag: Berlin, pp 104-117.

Dowsland KA (1997). Off-the-Peg or Made-to Measure? Timetabling and Scheduling with SA and TS. In Edmund Burke and Micheal Carter, editors, *The Practice and Theory of Automated Timetabling*. Lecture Notes in Computer Science 1408. Springer-Verlag: Berlin, pp 37-52.

Gendreau M and Potvin J (2005). Chapter 6: Tabu Search. In Edmund Burke and Graham Kendall, editors, *Search Methodologies: Introductory Tutorials in Optimisation and Decision Support Techniques*. Kluwer Academic Publishers: London

Glover F and Laguna M (1997). *Tabu Search*. Kluwer Academic Publishers: London.

Merlot LTG , Boland N, Hughes BD and Stuckey PJ (2003). A Hybrid Algorithm for the Examination Timetabling Problem. In Edmund Burke and Patrick De Causmaecker, editors, *The Practice and Theory of Automated Timetabling*. Lecture Notes in Computer Science 2740. Springer-Verlag: Berlin, pp 207-231.

Petrovic S and Burke EK (2004). University Timetabling. Ch. 45 in the *Handbook of Scheduling: Algorithms, Models, and Performance Analysis* (eds. J. Leung), Chapman-Hall/CRC Press

Schaerf A (1999^a). Local Search Techniques for High-school Timetabling Problems. *IEEE Transactions on Systems, Man, and Cybernatic* **29**(4), pp 368-377.

Schaerf A (1999^b). A Survey of Automated Timetabling. *Artificial Intelligence Review* **13**(2): 87-127.

Thompsom PM and Orlin JB (1989). *The Theory of Cyclic Transfer*. Working paper OR200-89, Operation Research Center, MIT, Cambridge, MA.

Thompsom PM and Psaraftis HN (1993). Cyclic Transfer Algorithm for Multi-vehicle Routing and Scheduling Problems. *Opns Res* **41**: 935-946.

White GM and Xie BS (2000). Examination Timetables and Tabu Search with Longer Term Memory. In Edmund Burke and Wilhelm Erbens, editors, *The Practice and Theory of Automated Timetabling*. Lecture Notes in Computer Science 2079. Springer-Verlag: Berlin, pp 85-103.

Wright M (2001). Subcost-Guided search – Experiments with Timetabling Problems. *Journal of Heuristics* **7**: pp 251-260.

```
Obtain a feasible initial solution Sol;
Calculate initial cost function f(Sol);
$Sol_{best} \leftarrow Sol$;
Create partition;
Define neighbourhood structures and construct the improvement graph, G;
Set not impoving counter $\leftarrow 0$;
do while (not termination-criterion)
    Find a negative cost partitiont-disjoint graph cycles for G using the modified
    shortest-path label correcting algorithm;
    Calculate the quality of a new solution f(Sol*);
    if ($f(Sol^*) \leq f(Sol_{best})$)
            $Sol \leftarrow Sol^*$;
            $Sol_{best} \leftarrow Sol^*$;
            not improving counter $\leftarrow 0$;
     else
           /* Generate exponential monte carlo acceptance criterion as below: */
           Calculate the difference between old and new solution, $\delta = f(Sol^*) - f(Sol))$;
           Generate RandNum, a random number in [0,1];
           if ($RandNum < e^{-\delta}$ )
                   $Sol \leftarrow Sol^*$;
                   not improving counter $\leftarrow 0$;
           else increase not improving counter by 1;
           if not improving counter == not_ improving_constant
                   exit;
     Add profitable or worse moves to the tabu list;
     Identify the cell for each move exams, given as AffectedCells;
     Calculate the number of AffectedCells, given as NumberOfAffectedCells;
     Recreate new cells from updated solution;
     Define neighbourhood structure and update G (see Figure 2);
end do;
```

Figure 1: Pseudo-code for the examination timetabling problem

```
Determine the cells that are involved in cyclic (or path) exchanges called
AffectedCells;
Determine the number of AffectedCells called NumberOfAffectedCells;
Keep the directed arcs that are not connected to / from the AffectedCells called
OriginalArcs;
Case 1:
    repeat
            Generate the directed arcs for every pair of exams from AffectedCells to
            other cells iff any of these pair of exams are not in the TabuList called
            TNewDirArcs1;
             Calculate the costs for the TNewDirArcs1;
    until (NumberOfAffectedCells)

Case 2:
    repeat
            Generate the directed arcs for every pair of exams from other cells to
            AffectedCells iff any of these pair of exams are not in the TabuList called
            TNewDirArcs2 ;
             Calculate the costs for the NewArcs2;
    until T
Combine OriginalArcs, TNewDirArcs1 and TNewDirArcs2 to form a new G;
```

Figure 2. Pseudo-code for updating the improvement graph

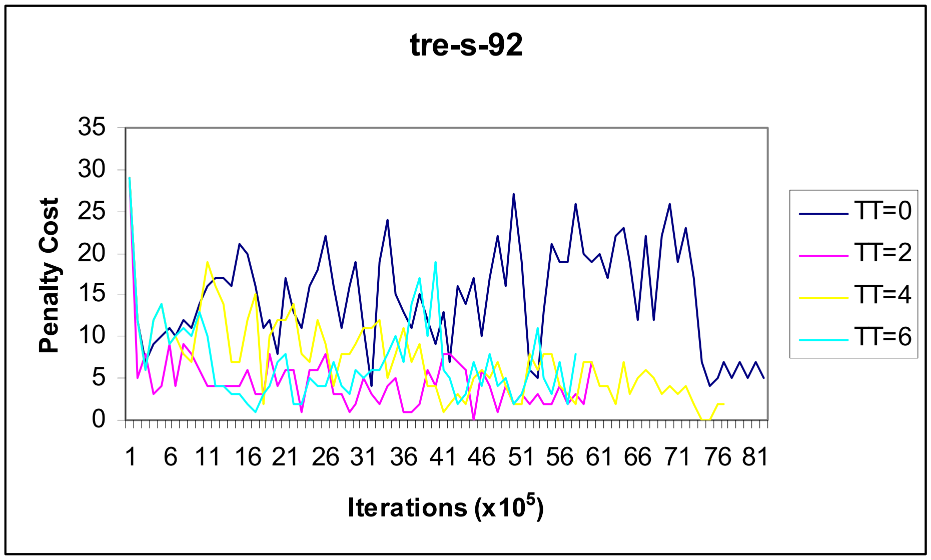

Figure 3: Performance of our algorithm with different TT on *tre-s-92* dataset

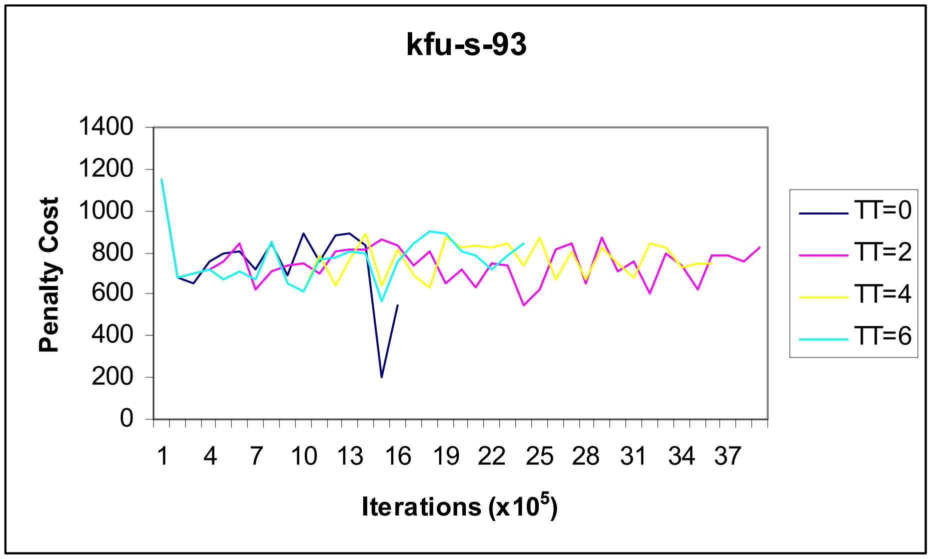

Figure 4: Performance of our algorithm with different TT on *kfu-s-93* dataset

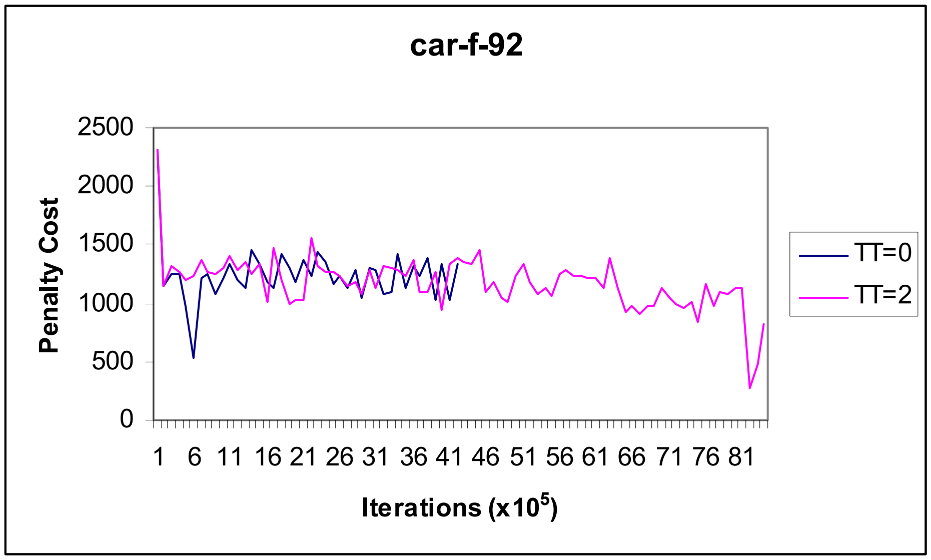

Figure 5: The behaviour of *car-f-92* dataset

Table 1: Capacitated Benchmarks

Data	Number of examinations	Number of Timeslots	Room Capacity	Conflict Matrix Density
car-f-92	543	31	2000	0.14
car-s-91	682	51	1550	0.13
kfu-s-93	461	20	1955	0.06
tre-s-92	261	35	655	0.18
uta-s-92	622	38	2800	0.13
nott-94	800	26	1550	0.03

Table 2: Results on Capacitated Problem

Data	Our tabu-based approach TT=2	TT=4	TT=6	Abdullah *et al*, 2006	Merlot *et al,* 2003	Di Gaspero and Schaerf, 2001	Caramia *et al,* 2001	Burke *et al,* 1996[a]
car-f-92	278	284	314	525	**158**	242	268	331
car-s-91	37	48	72	47	**31**	88	74	81
kfu-s-93	548	616	569	**206**	247	512	912	974
tre-s-92	**0**	**0**	1	4	**0**	4	2	3
uta-s-92	**300**	**300**	346	310	334	554	680	772
nott-94	18	21	32	-	**2**	11	44	53

Note: TT = Tabu tenure