AN INVESTIGATION OF VARIABLE NEIGHBOURHOOD SEARCH FOR UNIVERSITY COURSE TIMETABLING

SALWANI ABDULLAH[1], EDMUND K. BURKE[1] AND BARRY MCCOLLUM[2]

[1]*Automated Scheduling, Optimization and Planning Research Group, School of Computer Science & Information Technology, University of Nottingham, Jubilee Campus, Wollaton Road, Nottingham NG8 1BB, United Kingdom.*
{sqa@cs.nott.ac.uk,ekb@cs.nott.ac.uk}

[2]*Department of Computer Science, Queen's University Belfast, Belfast BT7 1NN, United Kingdom*
{b.mccollum@qub.ac.uk}

Abstract: The university course timetabling problem consists, in essence, of assigning lectures to a specific timeslot and room. The goal is to satisfy as many soft constraints as possible while constructing a feasible schedule. In this paper, we present a variable neighbourhood search approach with an exponential monte carlo acceptance criteria. This heuristic search approach is based on random-descent local search. The solution returned after exploring a neighbourhood structure is accepted based on the exponential monte carlo criteria. The approach is tested over eleven established datasets. The results demonstrate that the variable neighbourhood search approach is able to produce solutions that are competitive with state-of-the-art techniques from the literature.

Key words: Course Timetabling, Variable Neighbourhood Search

1. INTRODUCTION

Local search meta-heuristics are some of the most popular approaches in solving challenging optimisation problems including course timetabling (see, for example, Burke *et al*. 2003, Socha *et al*. 2002 and Di Gaspero and Schaerf 2002). It is beyond the scope of this paper to review the extensive literature on course timetabling but the interested reader can consult the following survey / overview papers (de Werra 1985, Schaerf 1999, Bardadym 1996, Burke *et al*. 1997, Carter and Laporte 1998, Burke and Petrovic 2002, Petrovic and Burke 2004 and Burke *et al*. 2004a). The success of finding good solutions for these problems is determined by the technique itself and the neighbourhood structure employed during the search. 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. The objective of this paper is to analyse the importance of different neighbourhood structures for course timetabling problem. We present and discuss a Variable Neighbourhood Search (VNS) approach to the problem.

In this paper, we consider the same course timetabling problem as described in Rossi-Dario *et al*. (2002) and Socha *et al*., (2002). The problem consists of:

- A set of N courses
- 45 timeslots (5 days with 9 timeslots each day)
- A set of R rooms
- A set of F room features
- A set of M students.

It deals with a set of courses $e = \{e_1,\ldots,e_N\}$ that need to be mapped to 45 timeslots subject to a variety of hard and soft constraints. Hard constraints should not be violated under any circumstances and we call a timetable while satisfies all such constraints a feasible solution. Socha *et al*. (2002) present the following hard constraints:

1. *No student can be assigned to more than one course at the same time.*
2. *The room should satisfy the features required by the course.*
3. *The number of students attending the course should be less than or equal to the capacity of the room.*
4. *No more than one course is allowed at a timeslot in each room.*

Socha *et al*. (2002) present the following soft constraints that are penalised equally.

1. *A student has a course scheduled in the last timeslot of the day.*

2. *A student has more than 2 consecutive courses.*
3. *A student has a single course on a day.*

The objective of this problem is to minimize the number of soft constraint violations (i.e. a penalty cost) in a feasible solution.

This paper is organised as follows: the basic algorithm of VNS is discussed in the next section. The implementation of VNS to course timetabling is discussed in Section 3 followed by a discussion on the experiments and results in Section 4. Brief conclusions are drawn in Section 5.

2. VARIABLE NEIGHBOURHOOD SEARCH (VNS)

VNS was introduced by Mladevonić and Hansen (1997). It is based upon the strategy of using more than one neighbourhood structure and of changing those structures systematically during the local search. This helps the VNS to explore neighbourhoods which are distant from the current solution and to jump to new solutions.

The basic VNS is made up of three stages: *shaking*, *local search* and *move*. More details can be found in Hansen and Mladenović (2001). A local search is applied repeatedly to obtain the local optima from the current solution (Mladenović and Hansen, 1997). Originally the basic VNS approach was a descent method. It does not accept a worsening solution to get out of local optima since the neighbourhood structures are varied regularly. Since the local optima in one neighbourhood structure is not necessarily a local optima in another neighbourhood structure, the change of the neighbourhood structures can be undertaken during the search. The termination criteria may be selected as a maximum number of iterations, the CPU time or a certain number of iterations without improvement.

Recently, research interest in the VNS approach has increased. For example Burke *et al.* (2004b) investigated VNS for nurse rostering problems where they employed two different search algorithms (steepest-descent and tabu search) implemented on a set of defined neighbourhood structures. Avanthay *et al*. (2003) implemented an adaptation of VNS for the graph colouring problem using the Tabucol algorithm (Hertz and Werra, 1987) as a local search employed on three neighbourhood structures (vertex, class and non-increasing neighbourhood). Other related works on VNS applied on a number of different problems can be found in Moreno Pérez *et al*. (2003), Mladenović *et al*. (2003), Hansen *et al*. (2001) and Caporossi and Hansen (2000).

3. VNS FOR COURSE TIMETABLING

This section discusses the applications of the basic VNS (referred to hereafter as VNS-Basic) and the modified VNS with exponential monte carlo acceptance criteria at the VNS level (referred to hereafter as VNS-EMC). The motivation for applying a monte carlo acceptance criteria is to better explore the search space.

3.1 Initial Solution: Constructive Heuristic

The initial solution is produced using a constructive heuristic which starts from an empty timetable. This feasible solution is obtained by adding or removing appropriate events (courses) from the schedule based on room availability (we attempt to schedule those courses with the least room

availabilities earlier on in the process), without taking into account any of the soft constraints, until the hard constraints are met. This constructive heuristic behaves likes a saturation degree heuristic (see Carter and Laporte, 1996). The schedule is made feasible before starting the algorithms.

3.2 Local Search

A local search heuristic explores the neighbourhood at the present solution by iteratively performing local changes in order to improve the quality of a solution until a local optima (the best solution(s) in the defined neighbourhood) is found. In this paper, we develop a random-descent local search that only accepts an improved solution after exploring the nearest neighbours in the defined neighbourhood structure.

3.3 Acceptance Criteria for VNS

In this work, we use two acceptance criteria to be applied at the VNS level. The motivation for applying acceptance criteria at the VNS level is to jump to other distant solution points by accepting worse solutions (i.e. case (b)).

a) Descent: It only accepts an improved solution.

b) Exponential monte carlo: This is quite similar to the acceptance criteria of simulated annealing but we do not need a cooling schedule. Exponential monte carlo is only based on the solution quality (no tuning parameters are needed). It accepts a worse solution with a certain probability. Let us denote the old and new solutions by s and s', respectively. The quality of the old and new solutions can be represented as $f(s)$ and $f(s')$, respectively. The new solution s' is accepted if a generated random number [0,1] is less than $e^{-\delta}$ where $\delta = f(s') - f(s)$. Increasing the value of δ will decrease the probability of accepting a worse solution. More details on the exponential monte carlo can be found in Ayob and Kendall (2003).

3.4 Neighbourhood Structures within VNS

We implemented the following neighbourhood structures to be used at the local search level:

(1) Move timeslot. Take 2 timeslots (selected at random), say t_i and t_j (where $j>i$) and the timeslots are ordered $t_1, t_2, \ldots, t_{45}$. Take all the exams in t_i and allocate them to t_j. Now take the exams that were in t_j and allocate them to t_{j-1}. Then allocate those that were in t_{j-1} to t_{j-2} and so on until we allocate those that were in t_{i+1} to t_i and terminate the process.

(2) Move the highest penalty course (i.e. a course with the highest number of soft constraint violations). Take 10% of the courses at random. Then select the one with the highest penalty cost and allocate it to the timeslot which generates the lowest penalty and which does not create an infeasibility.

(3) Move the highest penalty course from a random 30% selection of the courses to a random feasible timeslot.

(4) Move the highest penalty course from a random 20% selection of the courses to a random feasible timeslot.

(5) Move highest penalty course from a random 10% selection of the courses to a random feasible timeslot.
(6) Select a course at random and find another course at random which can swap timeslots.
(7) Choose single course at random and move to another random feasible timeslot.
(8) Move 2 courses to random feasible timeslots.
(9) Move 3 courses (as in (8)).
(10) Move 4 courses (as in (8)).
(11) Move 5 courses (as in (8)).
(12) Select two timeslots at random and simply swap all courses in one timeslot with all courses in the other timeslot.

The neighbourhood structure described in (1) is implemented before we apply the VNS approach. This neighbourhood structure allows courses to be moved relative to each other except for courses that are scheduled in the same timeslot.

3.5 The VNS Algorithm

For the approach presented in this paper, we apply the set of neighbourhood structures as presented in subsection 3.3. Hard constraints are never violated during the course of the timetabling process.

Let n_k (where $k = 1,\ldots,K$) be a set of predefined neighbourhood structures. Note that K is the total number of neighbourhood structures to be used in the search. Let $f(s)$ be the quality of the solution s. The local search starts by randomly generating a solution s' from the k^{th} neighbourhood. Starting from the initial solution s', the local search sequentially visits the k^{th} neighbourhood of s' until a local optima s'' is obtained. The solution s'' is accepted if $f(s'')$ is better than $f(s)$. This algorithm will also accept a worse solution with a certain probability that is generated from the exponential monte carlo acceptance criteria. Whenever a neighbourhood structure generates a better solution (or an accepted worse solution), the search starts over from the first neighbourhood. Otherwise, the next neighbourhood is employed. Based on our initial tests with the neighbourhood ordering for the course timetabling problem, we note that the best sequence of the neighbourhood structures is to order them in an increasing size. An increasing size-based neighbourhood structure ordering has also been experimented with in Burke et al. (2004b) where the authors claim that this ordering represents a strong approach in VNS when applied to nurse rostering problem. Figure 1 shows a schematic overview of the approach.

```
Initialisation:
(1)  Select the set of neighbourhood structure n_k, k=1,…,K
     that will be used in the random descent local search;
     find the initial solution s; choose a termination
     criteria;
(2)  Record the best solution s_best ← s and f(s_best) ← f(s);
Repeat until the termination criteria is met:
(1)  Set k ← 1;
(2)  Until k = K, repeat:
(a)  Shaking: Generate a point s' at random from the k^th
     neighbourhood of s (s'∈ n_k(s));
(b)  Local search: Apply a random-descent local search with
     s' as an initial solution until local optima s" is
     obtained.
(c)  Move or not: Accept s" (s ← s") if it is better than
     incumbent solution s. Set s_best ← s  and continue the
     search with n_1;
     otherwise,
        if f(s") is accepted by the acceptance criteria,
        then
           s ← s";
           Continue the search with n_1;
        otherwise
           Set k ←k+1
```

Figure 1: Pseudo code of a modified VNS for course timetabling problem

We further investigate the performance of VNS by adding a tabu list (Glover and Laguna 1993, 1997), to penalize neighbourhood structures that are not performed. We apply a tabu restriction where the neighbourhood structure will be tabu active if the new solution obtained is greater than the old solution and it is rejected by the exponential monte carlo acceptance criteria. The basic idea of the tabu list is to prevent a neighbourhood structure that did not perform well recently from being chosen in the next evaluations, so that we can direct the search to other possible areas of the search space. Based on our preliminary tests, the tabu tenure is set to be 2. A higher value of the tabu tenure is not appropriate in our case because we keep neighbourhood structures in the tabu list instead of moves. The higher the value of the tabu tenure, the longer the neighbourhood structures remain tabu, which will limit the number of the available neighbourhood structures to be used in the next evaluation in the VNS approach. The neighbourhood structures will remain tabu active for a number of evaluations which is equal to a tabu tenure. We call this approach VNS-Tabu. The pseudo code for VNS-Tabu is given in Figure 2.

```
Initialisation:
(1)  Select the set of neighbourhood structure n_k, k=1,…,K
     that will be used in the random descent local search;
     find the initial solution s; choose a termination
     criteria;
(2)  Record the best solution s_best ← s and f(s_best) ← f(s);

Repeat until the termination criteria is met:
(1)  Set k ← 1;
(2)  Until k = K, repeat:
(a)  Shaking: Generate a point s' at random from the k^th
     neighbourhood of s (s'∈ n_k(s));
(b)  Local search: Apply a random-descent local search with
     s' as an initial solution until local optima s" is
     obtained.
(c)  Move or not:
     if f(s") is better then incumbent solution s or f(s")
     is accepted by the acceptance criteria, then
        s ← s"
        set k ← 1;
        while k is in the tabulist
           k ← k+1;
        continue the search with n_k;
     otherwise
        insert k to the tabulist;
        set k ← k+1;
        while k is in the tabulist
           k ← k+1;
```

Figure 2: The pseudo code for VNS-Tabu

4 EXPERIMENTS AND RESULTS

The proposed method was tested on benchmark course timetabling problems from Socha *et al*. (2002) that are grouped into 5 small (N = 100, R = 5, F = 5 and M = 80) and 5 medium (N = 400, R = 10, F = 10 and M = 200) problems. We did not apply our VNS approach to the *large* dataset (N = 400, R = 10, F = 10 and M = 400) from Socha *et al*. (2002) since our constructive heuristic is unable to produce a feasible solution for this dataset. In Table 1, the test results on the benchmark course timetabling problems are presented for two types of variable neighbourhood search (i.e. VNS-Basic and VNS-EMC) "with and without ordering" of the neighbourhood structure. For the case of "without ordering", the VNS will continue the search with the current neighbourhood if it yields an improvement rather than going back to the first (set k = 1) neighbourhood structure (as in "with ordering" VNS) each time an improvement is found (or if the worse solution is accepted by the acceptance criteria). For instance, let k = 5 represent the current neighbourhood. The local optima obtained from this neighbourhood structure will be compared to the incumbent solution. If there is an improvement in the solution, then we repeat the search at k = 5. Otherwise we move to the next neighbourhood structure (represented by k = 6). This allows the next neighbourhood to be simply considered in the next search without needing to tune the sequence of the neighbourhood to determine

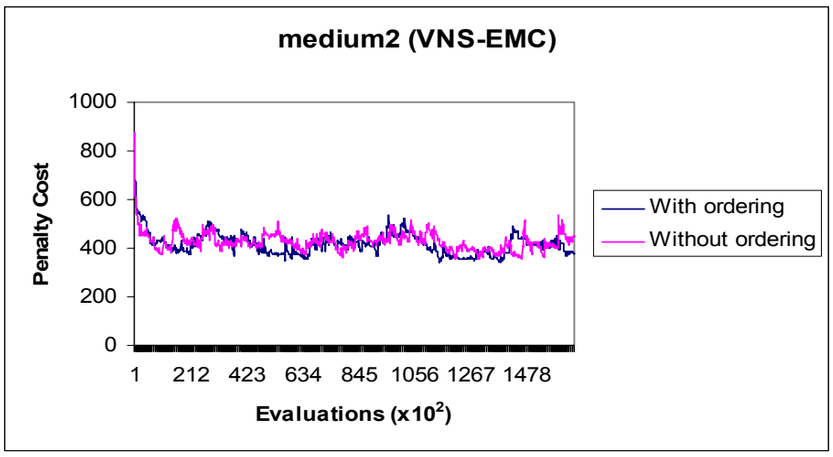
which neighbourhood will be employed. The motivation of this comparison is to see whether the sequence of neighbourhood structures plays a role in the VNS approach. We also make a comparison between the two VNS approaches to investigate the importance of accepting a worse move in order to jump to another part of the search space to obtain a better solution.

The approaches are coded in Microsoft Visual C++ version 6 under Windows. All experiments were run on an Athlon machine with a 1.2GHz processor and 256 MB RAM running under Microsoft Windows 2000 version 5. The number of evaluations for our approaches is 200000 (as in Socha *et al*., 2002) and the best results out of 5 runs obtained are presented.

Table 1: Results on course timetabling problem

Dataset	Initial Solution	With ordering VNS		Without ordering VNS	
		VNS-Basic	VNS-EMC	VNS-Basic	VNS-EMC
small1	261	8	0	19	1
small2	245	12	0	7	1
small3	232	8	0	11	1
small4	158	15	0	19	0
small5	421	5	0	5	0
medium1	914	418	338	445	347
medium2	878	414	337	413	354
medium3	941	441	384	462	390
medium4	865	381	299	406	324
medium5	780	390	307	416	315
large	100%Inf.	-	-	-	-

From Table 1, we can see that the “with ordering” VNS is better than or equal to the “without ordering” VNS for both VNS-Basic and VNS-EMC (except for *small2* and *medium2* datasets for VNS-Basic which is slightly higher). This shows that the ordering of the neighbourhood structure in the VNS approach and which neighbourhood the search will initiate after obtaining a better solution (or a worse accepted solution) is important. We believe this is because we give a chance for the search approach to explore by using different neighbourhood structures which can lead to different search spaces rather than letting the search approach employ the same neighbourhood structure until no more improvement on the solution quality is obtained (i.e. where the next neighbourhood structure only be employed on an unimproved solution). The results obtained from VNS-EMC are better than VNS-Basic for both the “with and without ordering” cases. This shows that accepting worse solutions (as a diversification strategy) helps the VNS approach to better explore the search space as suggested by Glover and Laguna (1997). Figures 3a and b show the behaviour of our approach applied to the *medium2* and *medium3* datasets on “with and without ordering” for VNS-EMC (to show the importance of ordering). Figures 4a and b show the behaviour of the VNS-Basic and VNS-EMC methods (for the case of “with ordering” VNS) on the *small4* and *small5* problems which indicates the importance of accepting worse solution in order to obtain a better final solution.



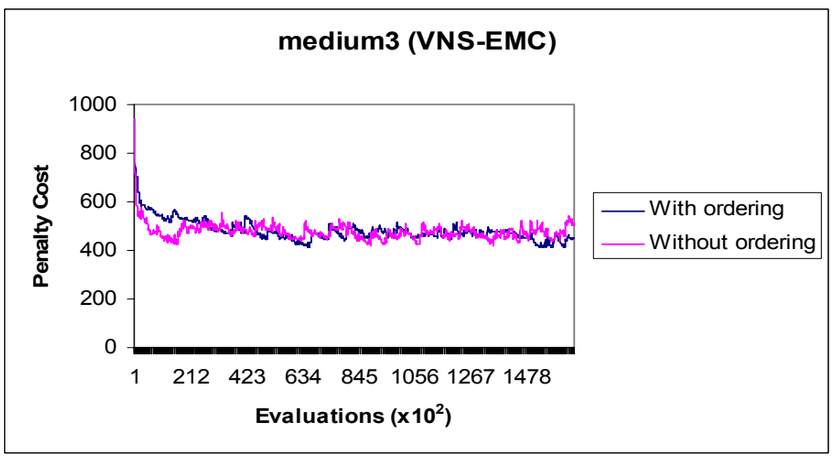
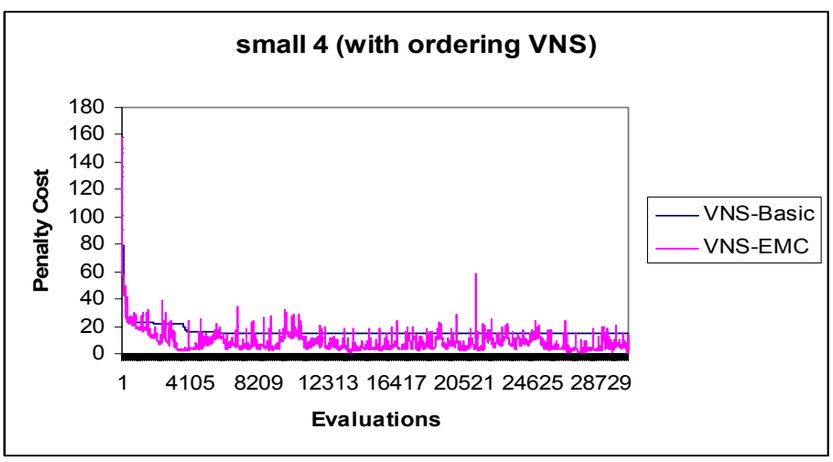


Figure 3a and b: The behaviour of *medium2* and *medium3* datasets, respectively



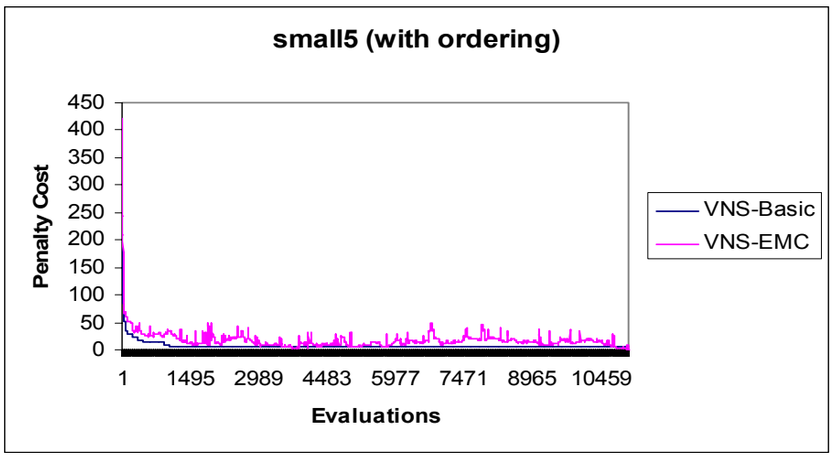


Figure 4a and b: The behaviour of *small4* and *small5* datasets, respectively

In all the figures, the x-axis represents the number of evaluations while the y-axis represents the penalty cost. Every point in the graphs corresponds to the penalty cost and number of evaluations of a separate solution. These graphs show how our algorithm explores the search space. In Figures 3a and b (and also for Figures 4a and b for VNS-EMC), the curves move up and down because we accept worse solutions with a certain probability in order to escape from local optima. The analysis of the graphs in Figures 3a and b show that the slope of the curves is relatively steep which indicates the high improvement in the quality of the solutions at the beginning of the search for both the "with and without ordering" cases of VNS-EMC. The improvement of the solution becomes slower as the search time increases but, with a good sequence of neighbourhood structures, the "with-ordering VNS" is able to find the better solution compared to the "without ordering VNS". This shows the importance of the sequence of neighbourhood. The graph in Figures 4a and b also illustrates the same behaviour as in Figures 3a and b at the early stage of the search with a high improvement in the solution quality. However, VNS-Basic gets stuck in the local optima after a certain number of evaluations. Whilst, by accepting worse solutions, the VNS-EMC is able to make further improvements and obtain a better solution (i.e. zero penalty cost). This indicates how a diversification strategy (i.e. accepting worse solutions) efficiently helps the technique to explore the search space.

Table 2 shows the comparison of our approach with other available approaches in the literature: A local search method and ant algorithm by Socha *et al*. (2002); a tabu-search hyperheuristic and a graph hyperheuristic by Burke *et al*. (2003) and Burke *et al*. (2005), respectively. Socha *et al*. (2002) present the average results out of 50 runs on the *small* problems, 40 runs on the *medium* problems and 10 runs on the *large* problem for their local search and ant algorithm. The term "x%Inf" in Table 2 indicates a percentage of runs that failed to obtain feasible solutions. The number of evaluations for both local search and the ant algorithm by Socha *et al*. (2002) was approximately 200000. The number of evaluations for the tabu-based hyperheuristic by Burke *et al.* (2003) was 12000 for the *small* problems, 1200 for the *medium* problems and 5400 for the *large* problem. Burke *et al*. (2005) did not note the number of evaluations in their approach.

Table 2: Comparison results on course timetabling problem

Dataset	VNS-EMC	Local search (Socha *et al*., 2002)	Ant Algorithm (Socha *et al*., 2002)	Tabu-based hyperheuristic (Burke *et al*., 2003)	Graph hyperheuristic (Burke *et al*., 2005)
small1	**0**	8	1	1	6
small2	**0**	11	3	2	7
small3	**0**	8	1	**0**	3
small4	**0**	7	1	1	3
small5	**0**	5	**0**	**0**	4
medium1	338	199	195	**146**	372
medium2	326	202.5	184	**173**	419
medium3	384	77.5% Inf	**248**	267	359
medium4	299	177.5	**164.5**	169	348
medium5	307	100% Inf	219.5	303	**171**
large	100%Inf	100% Inf	**851.5**	80% Inf 1166	1068

The best results are presented in bold. In terms of feasibility, we can see that our approach was able to produce ten out of eleven feasible solutions, whereas local search only produced nine feasible solutions (with two infeasible solutions for the *medium4* and *large* datasets). Other approaches are able to produce feasible solutions for all datasets. It can be seen that our approach produces better or equivalent results on *small* datasets when compared against all the other methods. Indeed, it is the only approach which can get zero penalty solutions on all 5 *small* problems. For the *medium* datasets, our results are comparable to the other results published in the literature except on the *medium3* and *medium5* datasets, where our method performs quite poorly. We believe that our approach performs well for *small* datasets because we might have more feasible solution points in our search space compared to *medium* datasets. We believe that we should be more selective in choosing the neighbourhood structures while solving the *medium* datasets.

Table 3 shows the comparison of the results obtained from VNS-Tabu, VNS-EMC and the best known results.

Table 3: The results on VNS-Tabu

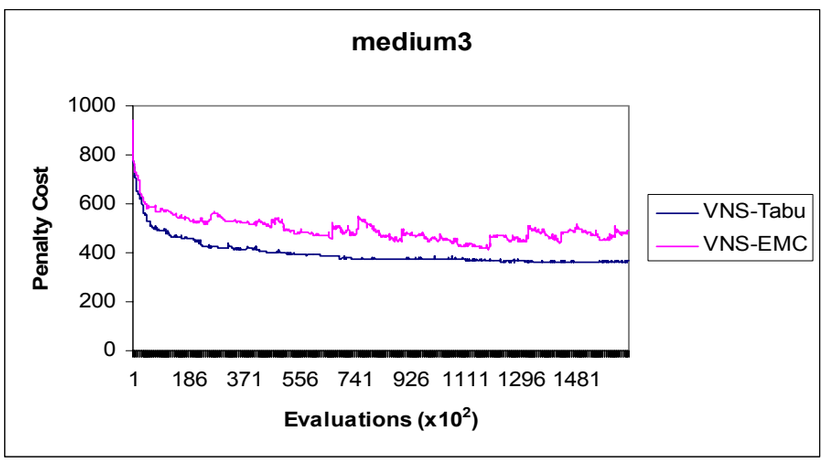
Dataset	VNS-Tabu	VNS-EMC	Best known
small1	**0**	**0**	1
small2	**0**	**0**	2
small3	**0**	**0**	**0**
small4	**0**	**0**	1
small5	**0**	**0**	**0**
medium1	*317*	338	**146**
medium2	*313*	326	**173**
medium3	*357*	384	**248**
medium4	*247*	299	**164.5**
medium5	*292*	307	**171**
large	100% Inf	100%Inf	**851.5**

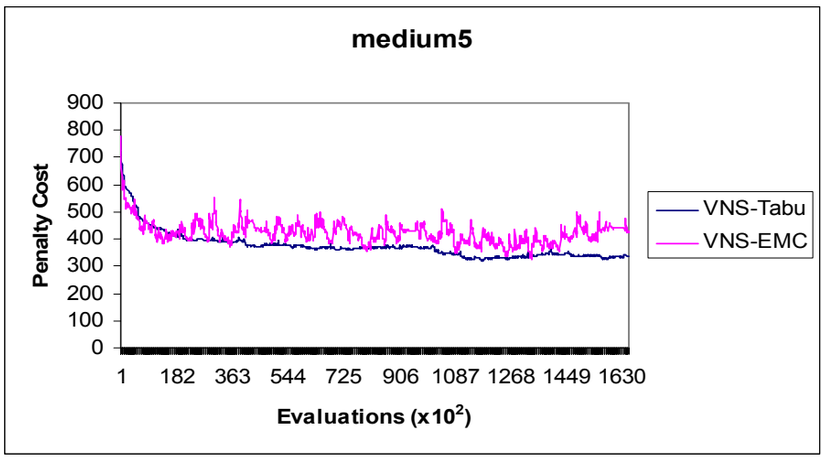
We are interested to compare the results from VNS-Tabu with our previous results to show that the tabu list can help to minimize the objective function for this problem. The results in Table 3 show the best result obtained from five runs on the VNS-Tabu. It can be seen that there is no difference between VNS-Tabu and VNS-EMC in terms of the penalty cost when applied to the *small* datasets. However, there are slight changes in the penalty cost for the *medium* datasets (as shown in italics). The percentage improvement (Δ %) obtained by applying the VNS-Tabu compared to VNS-EMC for the *medium* dataset is computed as:

$$\Delta\ \% = (\text{best VNS-EMC} - \text{best VNS-Tabu}) * 100 / \text{best VNS-EMC}$$

This shows that the VNS-Tabu managed to reduce the penalty cost between 4.0% to 17.4%. Although we cannot beat the best known results for the *medium* datasets, we are able to repair our results especially for the *medium3* and *medium5* datasets. Figures 5a and b show the behaviour of the VNS-Tabu when applied to the *medium3* and *medium5* datasets.

We use the same representation as in Figures 3 and 4 where the x-axis represents the number of evaluations while the y-axis represents the penalty cost. They show that the penalty cost can be quickly reduced at the beginning of the search where there appears to be a lot of room for improvement and this becomes less pronounced towards the end of the search (as in Figures 3a and b). It can be seen from Figure 5a that the VNS-Tabu is better compared to VNS-EMC throughout the search. However, in Figure 5b, the VNS-EMC is slightly better than VNS-Tabu in the early part of the search. Note that, by prolonging the search process we can see that the pattern of the VNS-Tabu is similar for both *medium3* and *medium5* datasets which show that the VNS-Tabu is able to find a good solution if given more search time. We believe that with the help of the tabu list, the VNS-Tabu performs better than VNS-EMC and we are able to find a better solution because the neighbourhood structures that are not used will not be employed in the next evaluations (the algorithm will only be supplied with the neighbourhood structures that are currently used) unlike the VNS-EMC approach.





Figures 5a and b: The behaviour of VNS-Tabu applied to *medium3* and *medium5* datasets, respectively

5 CONCLUSIONS

The overall goal of this paper was to investigate a modified VNS approach for the course timetabling problem. Preliminary comparisons indicate that our VNS approach is competitive with other approaches in the literature and is particularly suitable for *small* problems. On the five (out of eleven) *small* benchmark problems, the VNS approach produced solutions that were better than or equal to those published results (obtaining zero penalty cost for all five problems). Our future work will try to intelligently select a subset of neighbourhood structures from a much larger pool of neighbourhoods.

Acknowledgement

This work has been supported by the Public Services Department of Malaysia (JPA) and the University Kebangsaan Malaysia (UKM).

References

Ahuja, R.K., Orlin, J.B. and Sharma, D. (2000) Very large scale neighbourhood search. *International Transaction in Operational Research* 7, pp 301-317.

Avanthay, C., Hertz, A. and Zufferey, N (2003) A variable neighbourhood search for graph colouring. *European Journal of Operational Research* 151, pp 379-388.

Ayob, M. and Kendall, G. (2003) A monte carlo hyper-heuristic to optimise component placement sequencing for multi head placement machine. *Proc. Of the International Conference on Intelligent Technologies, InTech'03*, Chiang Mai, Thailand, 17-19 Dec, pp 132-141.

Bardadym, V. (1996) Computer-Aided School and University Timetabling: The New Wave. In Edmund Burke and Peter Ross, editors, *The Practice and Theory of Automated Timetabling I: Selected papers from the 1st International Conference, Springer Lecture Notes in Computer Science* Vol 1153, pp 22-45.

Burke EK, Jackson KS, Kingston JH and Weare RF (1997) Automated Timetabling: The State of the Art. *The Computer Journal*, Vol. 40, No. 9, pp 565-571

Burke EK and Petrovic S (2002) Recent Research Direction in Automated Timetabling. *European Journal of Operational Research* 140, pp 266-280.

Burke, E.K., Kendall, G. and Soubeiga, E. (2003) A tabu search hyperheuristic for timetabling and rostering. *Journal of Heuristics* 9(6), pp 451-470.

Burke EK, Kingston J and de Werra D (2004a) Applications to Timetabling. Handbook of Graph Theory, (editors, J. Gross and J. Yellen), Chapman Hall/CRC Press, pp 445-474.

Burke, E. K., De Causmaecker, P. and Vanden Berghe, G.(2004b) *Novel metaheuristic Apporoaches to Nurse Rostering Problems in Belgian Hospitals*, Ch. 44 in J. Leung: Handbook of Scheduling: Algorithms, Models and Performance Analysis, CRC Press, pp 44.1-44.18.

Burke, E.K., Meisels, A., Petrovic, S. and Qu, R. (2005) A graph-based hyper heuristic for timetabling problems. Accepted for publication in the *European Journal of Operational Research.*

Caporossi, G. and Hansen, P. (2000) Variable neighbourhood search for external graphs: I The AutoGraphiX system. *Discrete Mathematics* 212, pp 29-44.

Carter, M.W. and Laporte, G. (1996) Recent developments in practical examination timetabling. In Edmund Burke and Michael Carter, editors, *The Practice and Theory of Automated Timetabling I: Selected Papers from the 1st International Conference, Springer Lecture Notes in Computer Science* Vol 1153, pp 3-21.

Carter, M.W. and Laporte, G. (1998) Recent developments in practical course timetabling. In Edmund Burke and Peter Ross, editors, *The Practice and Theory of Automated Timetabling II: Selected Papers from the 2nd International Conference, Springer Lecture Notes in Computer Science* Vol 1408, pp 3-19.

Di Gaspero, L. and Schaerf, A. (2002) Multi-neighbourhood local search with application to course timetabling. In Edmund Burke and Patrick De Causmaecker, editors, *The Practice and Theory of Automated Timetabling IV: Selected Papers from the 4th International Conference, Springer Lecture Notes in Computer Science* Vol 2740, pp 262-275.

de Werra D. (1985) An Introduction to Timetabling. *European Journal of Operations Research*, 19, pp 151-162.

Glover and Laguna (1997) Tabu search. *Kluwer Academic Publisher*, ISBN 0-7923-8187-4.

Glover, F and Laguna, M. (1993) Tabu Search. In Reeves, C.R. (eds.) Modern Heuristic Technique for Combinatorial Problems. *Scientific Publications*, Oxford.

Hansen, P. and Mladenović, N. (2001) Variable neighbourhood search. *European Journal of Operational Research* 130, pp 449-467.

Hansen, P., Mladenović, N. and Perez-Britos, D. (2001) Variable neighbourhood decomposition search, *Journal of Heuristics* 7, pp 335-350.

Hertz, A. and de Werra, D. (1987) Tabu search techniques for graph colouring. *Computing* 39, pp 345-351.

Mladenović, N. and Hansen, P. (1997) Variable neighbourhood search. *Computers and Operations Research* 24(11), pp 1097-1100.

Mladenović, N., Petrović J., Kovačevič-Vujčić, V. and Čangalović, M. (2003) Solving spread spectrum radar polyphase code design problem by tabu search and variable neighbourhood search. *European Journal of Operational Research* 151, pp 389-399.

Morena Pérez, J.A., Marcos Moreno-Vega, J. and Rodríguez Martín, I. (2003) Variable neighbourhood tabu search and its application to the median cycle problem, *European Journal of Operational Research* 151, pp 365-378.

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.

Rossi-Doria, O., Blum, C., Knowles, J., Samples, M., Socha, K. and Paechter, B. (2002) A local search for the timetabling. *Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling (PATAT)*, pp 124-127.

Schaerf A (1999) A Survey of Automated Timetabling. Artificial Intelligence Review 13(2), pp 87-127.

Socha, K., Knowles, J. and Samples, M. (2002) A max-min ant system for the university course timetabling problem. *Proceedings of the 3rd International Workshop on Ant Algorithms, ANTS 2002, Springer Lecture Notes in Computer Science* Vol 2463 (10), pp 1-13.

Thompson, J. and Dowsland, K. (1996) General cooling schedule for a simulated annealing based timetabling system. In Burke, E.K. and Ross, P. (eds). *Practice and Theory of Autamated Timetabling: Selected Papers from the 1st International Conference. Springer Lecture Notes in Computer Science* Vol 1153, pp 345-363.

Thompson, J. and Dowsland, K. (1998) A robust simulated annealing based examination timetabling system. *Computers and Operations Research* 25, pp 637-648.