An Iterative Re-start Variable Neighbourhood Search for the Examination Timetabling Problem

Masri Ayob[12], Edmund K. Burke[1] and Graham Kendall[1]

[1] ASAP Research Group, School of Computer Science & IT
University of Nottingham, Nottingham NG8 1BB, UK
{ekb, gxk}@cs.nott.ac.uk

[2] Faculty of Information Science and Technology
Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia
masri@ftsm.ukm.my

1 Introduction

Producing good quality examination timetables is a difficult task which is faced by many academic institutions. Due to the complexity and the large size of the real-world university examination timetabling problems, it is difficult to obtain an optimal solution. Indeed, due to the complex nature of the problem, it is questionable if an end user would recognise a truly optimal solution.

Carter and Laporte [1] defined the exam timetabling as: "the assigning of examinations to a limited number of available time periods in such a way that there are no conflicts or clashes."

In principle, the exam timetabling problem involves, assigning exams to timeslots subject to a set of hard and soft constraints. Hard constraints are rigidly enforced whilst soft constraints should be satisfied as far as possible. For example, exams which have common students have to be assigned to different timeslots (hard constraint). Wherever possible, examinations should be spread out over timeslots so that students have large gaps in between exams (soft constraint). Constraints vary among institutions and further discussion of exam timetabling constraints can be found in Burke et al. [2] and Carter and Laporte [1]. Timetables that satisfy all the hard constraints are called feasible solutions. Due to the complexity of the problem, it is not usually possible to have solutions that do not violate some of the soft constraints. Indeed, the evaluation of the cost function (how good the solutions are) is a function of violated soft constraints. A weighted penalty value is associated with each violation of the soft constraint and the objective is to minimise the total penalty value.

The exam timetabling problem can be modelled as a graph colouring problem (see Burke et al. [3, 4]). Usually, graph colouring heuristics, which order the events/exams based on an estimation of their difficulties, are used to construct the timetable. These include:

Largest degree first. This first schedules the exam that has the largest number of conflicts with other exams.

E. K. Burke, H. Rudová (Eds.): PATAT 2006, pp. 336–344. ISBN 80-210-3726-1.

Colour degree. Exams with a greater number of conflicts with the exam that have already been scheduled have a higher priority of being scheduled next.

Saturation degree. Exams with fewer feasible slots are scheduled as early as possible.

Largest weighted degree. Exams with the higher number of students in conflict are scheduled earlier.

Largest enrolment. Exams with larger student enrolments are scheduled earlier.

Many approaches have been developed to solve the exam timetabling problem. These include graph heuristics (Burke et al. [5]), tabu search (Di Gaspero and Schaerf, [6]), evolutionary algorithms (Burke et al. [7]; Erben [8]; Côté et el. [9]), simulated annealing (Dowsland [10]), hyper-heuristics (Burke et al.[4]) etc. Extensive surveys and overviews on various approaches in solving timetabling problem can be found in Carter [11], Carter and Laporte [1], Schaerf [12], Burke and Petrovic [13] and Petrovic and Burke [14].

A survey by Carter [11] covers the exam timetabling research from 1964 until 1984, with Carter and Laporte [1] extending that survey. The later survey classifed the methods used in solving the exam timetabling problem into four groups: cluster, sequential, meta-heuristic and constraint-based approaches. Cluster methods group the exams and then assign a timeslot to each group. Sequential approaches assign exams to timeslots consecutively. Constraint-based methods represent exams as a set of variables that have to be assigned to which values that represent resources such as rooms and timeslots, which satisfy some constraints (White [14]). Meta-heuristic approaches start with initial solution(s) and then apply search strategies to improve the solutions. Carter and Laporte [1] argued that most approaches only use simple constraints in solving the exam timetabling problems.

The exam timetabling problem is considered as an un-capacitated problem when room capacity is ignored. Whereas, a capacitated problem limits the number of students sitting exams in a slot but does not directly assign exams to specific rooms. The benchmark datasets (available at ftp://ftp.mie.utoronto.ca/pub/carter/testprob), which are used in this work, are an un-capacitated problem. These benchmark datasets were presented by Carter et al. [15].

This paper describes a computational approach to examination timetabling. A constructive heuristic based on saturation degree is used, followed by a local improvement heuristic based on a variant of variable neighbourhood search.

2 Variable Neighbourhood Search

Variable neighbourhood search (VNS) was introduced by Mladenović and Hansen [16]. VNS is a heuristic that is capable of exploring multi-neighborhood structures, hence it can explore distant neighbourhoods of the current solution (Mladenović and Hansen [16]). The shaking procedure in the basic VNS approach is a diversification factor whilst the local search will intensify the search to lead it converge to a local optimum. A local search (see Reeves and Beasley [17]; Aarts and Lenstra [18]) is applied repeatedly to obtain the local optima from the selected neighbouring solution.

There has recently been increasing interest in the *VNS* approach. For example, Avanthay et al. [19], developed an adaptation of *VNS* to solve the graph colouring problem with a Tabucol (a variant of tabu search) algorithm (Hertz and de Werra [20]) as a local search. They used three neighbourhood structures; these being vertex, class, and non-increasing neighbourhoods. Their VNS algorithm, however is not superior to the hybrid algorithm proposed by Galinier and Hao [21] that integrates a tabu search and a genetic algorithm. Some other works on VNS include Caporossi and Hansen [22], Morena Pérez et al. [23] and Fleszar and Hindi [24], which demonstrate that it is suitable across a number of different problem types.

3 Iterative Re-start Variable Neighbourhood Search

The basic *VNS* algorithm is a descent heuristic (Hansen and Mladenović, [25]), whilst our iterative re-start *VNS (IR-VNS)* is a descent-ascent heuristic. Let n_w, $w=1,2\ldots,W$, be a set of predefined neighbourhood structures, and $n_w(x)$ is the set of solutions in the w^{th} neighbourhood of x, $f(x)$ is the quality of solution x. W is the total number of neighbourhood structures to be used in the search. Our *VNS* algorithm is presented in fig. 1.

In our approach, we do not apply a shaking procedure before starting the local search since this might prolong the search time. Since exam timetabling problems have to deal with many constraints, finding good quality feasible solutions can be difficult and time consuming. The shaking mechanism within basic *VNS* (Mladenović and Hansen [16]) may cause the search to jump to a poor solution which may be difficult to escape from. Therefore, in our approach, we replace the shaking procedure by accepting the best neighbour (which might be worse) of the incumbent solution.

Since the initialisation strategy could influence the performance of the search algorithm (see Burke et al., [26]), especially when the search space is disconnected (which is a common case in exam timetabling problem), we use the saturation degree heuristic to iteratively construct incumbent solutions when the search becomes trapped in a local optimum.

At the improvement stage, we employ four neighbourhood structures as follows:

1. Steepest descent (N_1). A neighbour solution of x is generated by swapping all exams in the j^{th} slot with all exams in the $(j+k)^{th}$ slot. This local search returns the best neighbour after visiting all neighbours of x.
2. Free slot (N_2). A neighbour solution of x is generated by changing the slot of each exam to the best available slot. The local search returns the best neighbour (not necessarily an improved solution) after assigning the best new slot for each exam.
3. Swap two exams (N_3). A neighbour solution of x is generated by swapping one exam in the j^{th} slot with one exam in the $(j+k)^{th}$ slot. Again, this local search returns the best neighbour after visiting all neighbours of x.

Exponential Monte Carlo, EMCQ (N_4). This local search is adapted from Ayob and Kendall [27]. The EMCQ accepts an improved solution but probabilistically accepts a worse solution depending on the solution quality, search time and the time it is trapped in a local optimum. As in N_1, a neighbour solution of x is generated by swapping all exams in the j^{th} slot with all exams in the $(j+k)^{th}$ slot. However, a trial solution will be accepted based on the EMCQ acceptance criterion. If it is accepted,

Step A: (Initialisation)
- *(1) Select the set of neighbourhood structures n_w, w=1,2…,W that will be used in the search; choose a stopping condition and value of MAX;*
- *(2) Sorts the exams in decreasing number of exams in conflicts; Let N as the number of exams and E_i as the i^{th} exam where $i \in \{1,2,...,N\}$. Set i=1;*

Step B: (Construction Stage)
- *(1) Use E_i as a starting exam and construct an incumbent solution x using saturation degree heuristic with back-tracking and random slot assignment;*
- *(2) If this is the first iteration, then record the best obtained solution, $x_{best} \leftarrow x$ and $f(x_{best}) \leftarrow f(x)$;*
- *(3) Set Unimproved=0;*

Step C: (Improvement Stage)
- *(1) Set $w \leftarrow 1$;*
- *(2) Do*
  - *a). Exploration of neighbourhood. Find the best neighbour, x' from the w^{th} neighbourhood of $x(x' \in n_w(x))$.*
  - *b). Accept the solution, $x \leftarrow x'$;*
  - *c). If $f(x') < f(x_{best})$ then $x_{best} \leftarrow x'$, $f(x_{best}) \leftarrow f(x')$ and Unimproved=0;*
  - *d) Else, Unimproved= Unimproved+1;*
  - *e) $w \leftarrow (w \bmod W)+1$;*

  Until Unimproved =MAX or the stopping condition is met.
- *(3) If Unimproved =MAX but the stopping condition is not met, then $i=(i \bmod N)+1$ and goto step B.*
- *(4) Otherwise, return the best obtained solution.*

Note: MAX is the upper bound for Unimproved counter.

Fig. 1. The Proposed VNS algorithm for exam timetabling problem

we move to a new solution and the search continues by exploring a new neighbourhood (by continuing with the subsequent slot). Some moves might be performed before returning the best obtained solution to the VNS level. This is different from the N_1 neighbourhood, which only returns the best neighbour of one neighbourhood.

Currently, we use the following permutation: $\{N_1, N_2, N_1, N_3, N_1, N_4\}$. We repeatedly apply N_1 after exploring each neighbourhood structure because N_1 explores all neighbours of one neighbourhood. Since each local search explores different

neighbourhood structures, it might be worth visiting all neighbours in N_1 after applying other local searchers. We are still investigating the effectiveness of repeatedly applying N_1, the order in which neighbourhood are explored and the upper bound value, *MAX* for an *Unimproved* counter. At this stage, we use *MAX*=20 and the above permutation.

4 Objective Function

We use a proximity cost as an objective function, which has been presented in Carter et al. [15], as follows:

$$\textbf{Minimise} \quad F = \frac{\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{ij} . proximity \ (t_i, t_j)}{M} \tag{1}$$

Subject to:

$$\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} c_{ij} . x(t_i, t_j) = 0 \tag{2}$$

$$x(t_i, t_j) = \begin{cases} 1 & \text{if} \quad t_i = t_j; \\ 0 & \text{otherwise;} \end{cases} \tag{3}$$

Where,

N: *is a number of exams;*

M: *is a number of students;*

T: *is a given number of available timeslot;*

$C = (c_{ij})_{NxN}$ *: is a conflict matrix where each element denoted by* c_{ij}*, where* $i,j \in \{1,2,...,N\}$*, is the number of students taking exam i and j;*

t_i *: is a timeslot for exam i where* $t_i \in \{0,1,\ldots,T-1\}$

$$proximity \ (t_i, t_j) = \begin{cases} \dfrac{2^5}{2^{|t_i - t_j|}} & \text{if} \quad 1 \le |t_i - t_j| \le 5; \\ 0 & \text{otherwise;} \end{cases} \tag{4}$$

Equation 4 presents a weighted value (suggested by Carter et al. [15]) that reflect the cost of assigning exam *i* and *j* to timeslots. These being 0, 1, 2, 4, 8 and 16, where the cost is '0' if the gap of slot for exam *i* and *j* is greater than 5. Equation (2) and (3) ensure a clash free timetable where each student will be sitting one exam at each timeslot.

5 Experiments and Results

In this work, we use the benchmark exam timetabling datasets presented in Carter et al. [15]. These datasets have been used by many researchers. However, due to some changes made by Carter et al. [15], there are two sets of benchmark dataset. Table 1 shows the latest version (updated on June 7, 2005) of Carter et al. [15].

Table 1. Characteristics of benchmark exam timetabling problems.

	Exams	**Students**	**Slots**
car-f-92	543	18,419	32
car-s-91	682	16,925	35
ear-f-83	190	1,125	24
hec-s-92	81	2,823	18
kfu-s-93	461	5349	20
lse-f-91	381	2,726	18
pur-s-93	2419	30,032	43
rye-f-92	486	11,483	23
sta-f-83	139	611	13
tre-s-92	261	4,360	23
uta-s-92	622	21,266	35
ute-s-92	184	2,750	10
yor-f-83	181	941	21

As discussed in section 3, this is ongoing research. Our preliminary experiment shows the following results (see table 2), which are comparable to the state-of-the-art approaches reported in the literature (Abdullah et al. [28]; Asmuni et al., 2005; Burke et al., [5, 4]; Burke and Newall [29]; Caramia et al.[30]; Carter et al. [15]; Di Gaspero and Schaerf, [6]).

Based on our preliminary result in table 2, we can see that our *IR-VNS* is capable of producing good quality solutions across all datasets. This shows that exploring various neighbourhood structures with iterative re-start is an effective search, which can avoid local optima and can jump to distant neighbourhood that might be more promising region.

Table 2. Results from our IR-VNS and the state-of-art approaches on benchmark exam timetabling problems based on the proximity cost..

	IR-VNS	**Abdullah et al. [28]**	**Asmuni et al. [31]**	**Burke et al. [5]**	**Burke et al. [4]**	**Burke and Newall [29]**	**Caramia et al. [30]**	**Carter et al. [15]**	**Di Gaspero and Schaerf [6]**
car-f-92	4.51	4.36	4.56	4.2	4.84	**4.0**	6.0	6.2	5.2
car-s-91	4.90	5.21	5.29	4.8	5.41	**4.6**	6.6	7.1	6.2
ear-f-83	36.28	34.87	37.02	35.4	38.19	37.05	**29.3**	36.4	45.7
hec-s-92	11.06	10.28	11.78	10.8	12.72	11.54	**9.2**	10.8	12.4
kfu-s-93	14.74	**13.46**	15.81	13.7	15.76	13.9	13.8	14.0	18.0
lse-f-91	12.08	10.24	12.09	10.4	13.15	10.82	**9.6**	10.5	15.5
pur-s-93	4.66	-	-	4.8	-	-	-	3.9	-
rye-f-92	10.67	8.74	10.35	8.9	-	-	-	7.3	-
sta-f-83	157.32	159.20	160.42	159.1	**141.08**	168.73	158.2	161.5	160.8
tre-s-92	8.92	**8.13**	8.67	8.3	8.85	8.35	9.4	9.6	10.1
uta-s-92	3.58	3.63	3.57	3.4	3.54	**3.2**	3.5	3.5	4.2
ute-s-92	26.36	**24.21**	27.78	25.7	32.01	25.83	24.4	25.8	29.0
yor-f-83	38.97	**36.11**	40.66	36.7	40.13	36.8	36.2	41.7	41.0

6 Conclusions

We have presented an iterative re-start variable neighbourhood search that has two stages; construction and improvement. In the construction stage, we employ a saturation degree graph colouring heuristic with back-tracking to construct an incumbent solution. We apply a variant of variable neighbourhood search to improve the incumbent solution. In our approach, we do not apply a shaking procedure before starting the local search. However, we diversify the search by accepting the best neighbour returned by the local search (which is not necessarily an improved solution). When the search becomes trapped in a local optimum, we jump to a distant solution space by reconstructing the incumbent solution using the saturation degree heuristic. Our preliminary results shows that our strategy is very promising, which can produce good quality solutions that are comparable to other published result.

References

1. Carter, M.W. and Laporte, G., (1996). Recent developments in practical examination timetabling. In: Burke and Ross (1996), 3-21.
2. Burke, E.K., Elliman, D.G., Ford, P. and Weare, R. F. (1996). Examination timetabling in British Universities – A survey. In: Burke and Ross, 76-92.
3. Burke, E.K., Elliman, D.G. and Weare, R.F., (1994). A university timetabling system based on graph colouring and constraint manipulation. Journal of Research on Computing in Education, 27(1), 1-18.

4. Burke, E.K., McCollum, B., Meisels, A., Petrovic, S. and Qu, R. (2006). A graph-based hyper-heuristic for educational timetabling problems. European Journal of Operational Research, in press.
5. Burke, E.K., Kingston, J. and de Werra, D. (2004a). Applications to timetabling. In: Gross, J. and Yellen, J. (eds.), Handbook of graph theory. Chapman Hall/CRC Press, 445-474.
6. Di Gaspero, L. and Schaerf, A., (2001). Tabu search techniques for examination timetabling. In: Burke, E.K. and Erben, W. (eds.), Selected papers from the 3rd International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, 2079, 104-117.
7. Burke, E.K., Newall, J. and Weare, R. (1998). Initialization strategies and diversity in evolutionary timetabling. Evolutionary Computation, 6(1), 81-103.
8. Erben, W., (2001). A grouping genetic algorithm for graph coloring and exam timetabling. In: Burke, E.K. and Erben, W. (eds.), Selected papers from the 3rd International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, 2079, 132-158.
9. Côté, P., Wong, T. and Sabourin, R., (2005) A hybrid multi-objective evolutionary algorithm for the uncapacitated exam proximity problem. In: Burke, E.K. and Trick, M. (eds.), Selected papers from the 5rth International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, 3616, Springer-Verlag, 294-312.
10. Dowsland, K., (1998). Off the peg or made to measure. In: Burke, E.K. and Carter, M. (eds.), Selected papers from the 2nd International Conference on the Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science, 1408, 37-52.
11. Carter, M.W., (1986). A survey of practical applications of examination timetabling algorithms. Operations Research Society of America, Volume 34 No 2, 193-202.
12. Schaerf, A., (1999) A survey of automated timetabling. Artificial Intelligence Review, 13, 87-127.
13. Burke, E.K. and Petrovic, S. (2002). Recent research directions in automated timetabling. European Journal of Operational Research, 140, 266–280.
14. Petrovic, S. and Burke, E.K., (2004), ch. 45- University timetabling. In: The Handbook of Scheduling: Algorithms, Models, and Performance Analysis. Leung, J. (edd.), CRC Press.
14. White, G. M., (2000). Constrained satisfaction, not so constrained satisfaction and the timetabling problem. Proceedings of the 3rd International Conference on the Practice and Theory of Automated Timetabling, Konstanz, Germany, 32-47.
15. Carter, M. W., Laporte, G. and Lee, S. Y. (1996) Examination timetabling: algorithmic strategies and applications. Journal of the Operational Research Society Volume 47 Issue 3, 373-383.
16. Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. Computers and Operations Research, 24(11), 1097-1100.
17. Reeves, C.R. and Beasley, J.E. (1995). Introduction, ch. 1. In: Reeves, C. R.(eds) Modern heuristic techniques for combinatorial problems, McGraw-Hill, 1-19.
18. Aarts, E. and Lenstra, J.K. (eds). (2003). Local search in combinatorial optimization. Princeton University Press.
19. Avanthay, C., Hertz, A. and Zufferey, N. (2003). A variable neighborhood search for graph coloring. European Journal of Operational Research, 151, 379-388.
20. Hertz, A. and de Werra, D. (1987). Using tabu search techniques for graph coloring, Computing, 39, 345-351.
21. Galinier, P. and Hao, J.-K. (1999). Hybrid evolutionary algorithms for graph coloring. Journal of Combinatorial Optimization, 3, 379-397.

22. Caporossi, G. and Hansen, P. (2004). Variable neighborhood search for extremal graphs 5. Three ways to automate finding conjectures. Discrete Mathematics, 276, 81-94.
23. Morena Pérez, J.A., Marcos Moreno-Vega, J. and Rodríguez Martín, I. (2003). Variable neighborhood tabu search and its application to the median cycle problem. European Journal of Operational Research, 151, 365-378.
24. Fleszar, K. and Hindi, K.H., (2004). Solving the resource-constrained project scheduling problem by a variable neighbourhood search. European Journal of Operational Research, 155(2), 402-413.
25. Hansen, P. and Mladenović, N. (2001). Variable neighborhood search. European Journal of Operational Research, 130, 449-467.
26. Burke, E.K., Bykov, Y., Newall, J. and Petrovic, S., (2004b). A time-predefined local search approach to exam timetabling problems. IIE Transactions, 36, 509-528.
27. 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, 132-141.
28. Abdullah, S., Ahmadi, S., Burke, E.K. and Dror, M. (2004). Applying Ahuja-Orlin's Large Neighbourhood for Constructing Examination Timetabling Solution , Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling (PATAT), 413-419.
29. Burke, E. K. and Newall, J. P. (2003). Enhancing timetable solutions with local search methods. In: Practice and Theory of Automated Timetabling IV, E. K. Burke and P. De Causmaecker (Eds.), Lecture Notes in Computer Science Vol. 2740, Springer-Verlag, pp. 195-206.
30. Caramia, M., Dell' Olmo, P. and Italiano, G. F. (2001) New algorithms for examination timetabling. In: Algorithm Engineering 4th International Workshop, S. Näher and D. Wagner (Eds), Lecture Notes in Computer Science, Volume 1982, Springer-Verlag, 230-241.
31. Asmuni, H. and Burke, E.K. and Garibaldi, J.M., Fuzzy Multiple Heuristic Ordering for Course Timetabling, in Proceedings of the 5th United Kingdom Workshop on Computational Intelligence (UKCI05), pp. 302-309, London, UK, 5-7 September 2005