A Hybrid Approach for University Course Timetabling

Salwani Abdullah and Abdul Razak Hamdan

Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia

Summary
The university course timetabling problem deals with the assignment of lectures to specific timeslots and rooms. The goal is to satisfy the soft constraints to the largest degree possible while constructing a feasible schedule. In this paper, we present a hybrid approach consisting of three phases. During phase 1, initial solutions are generated using a constructive heuristic. An improvement approach is employed in phase 2 using a randomised iterative algorithm with a composite neighbourhood structure and a simulated annealing based acceptance criterion. In phase 3, a hill climbing approach is implemented in an attempt to further improve the solution. The approach is tested on eleven established datasets. The results demonstrate that the hybrid approach is able to produce solutions that are competitive with state-of-the-art techniques from the literature.

Key words: Course Timetabling, Hybrid Approach, Composite neighbourhood structure, Hill Climbing.

1. Introduction

Several review papers discuss the major approaches which have been adopted in relation to educational timetabling (see de Werra 1985, Carter 1986, Carter and Laporte 1998, Bardadym 1996, Burke et al. 1997, Schaerf 1999, Burke and Petrovic 2002 and Petrovic and Burke 1998). In this paper a hybrid approach is proposed and implemented. The aim of the hybrid approach is to take the best ideas from one approach and incorporate them with other good (or better) ideas from other approaches. Hybridisation has proven to be very effective in the examination timetabling literature; for example, Caramia et al. (2001) obtained the best known results on several of the benchmark instances. The authors used improvement steps after employing a greedy scheduler that assigns examinations (which are ordered based on the degree of conflict) in turn to the lowest available timeslot with respect to the conflict free requirement.

Merlot et al. (2003) applied a hybrid method that consists of constraint programming, simulated annealing and hill climbing for uncapacitated and capacitated examination timetabling and obtained the best known results on some benchmarks. Constraint programming is used to generate feasible initial solutions and the quality of the timetable is improved using simulated annealing where a kempe chain neighbourhood (Thompson and Dowsland 1996, 1998) is employed. Then the hill climbing technique is utilised to further refine the timetable. Experimental results show that this method is superior to the method currently used by the University of Melbourne and performs well in comparison with other standard benchmark instances. Burke and Newall (2003) hybridised the approach in Burke and Newall (2004) with the great deluge method by Burke et al. (2004) to build a method which has the best known results on certain examination benchmark problems. Another example of a hybrid approach applied to the examination timetabling problem can be found in Côté et al. (2005). Kostuch (2005) employed a three-phase approach for the course timetabling problem which combined graph colouring and simulated annealing. In the first phase, an initial feasible timetable is generated using certain graph colouring heuristics. Improvement is made in the second phase using simulated annealing. Local search guided by simulated annealing is employed in the final phase to make further improvement.

This paper is organised as follows: the description on the university course timetabling problems is discussed in the next section. The implementation of the hybrid method 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. The University Course Timetabling Problem

The course timetabling problem deals with the assignment of a set of lecture events to a specific timeslot and room...
within a working week subject to a variety of hard and soft constraints. Hard constraints should not be violated under any circumstances and we call a timetable that satisfies all such constraints a feasible solution. In this paper, we consider the same course timetabling problem as described in Rossi-Dario et al. (2002) and Socha et al. (2002) who present the following hard constraints:

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

The following soft constraints, that are equally penalised, were also presented by Socha et al. (2002):

- A student has a course scheduled in the last timeslot of the day.
- A student has more than 2 consecutive courses.
- A student has a single course on a day.

The problem has

- A set of \( N \) courses, \( e = \{e_1, \ldots, e_N\} \)
- 45 timeslots
- 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. The objective of this problem is to satisfy the hard constraints and to minimise the violation of the soft constraints whilst constructing a feasible schedule.

3. A Hybrid Approach

In this paper, a hybrid approach is implemented which consists of 3 phases. In the first phase, the initial solution is generated using a constructive heuristic. The improvement technique, applied in the second phase, incorporates an iterative improvement algorithm with composite neighbourhood structures and uses simulated annealing as an acceptance criterion. A hill climbing technique is employed in the third stage to further enhance the quality of the timetable. The details of each phase are discussed below.

Phase 1: Constructive heuristic

The initial solution is produced using a constructive heuristic which starts with an ‘empty’ timetable. A 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. Thus the schedule is made feasible before starting on the improvement phase.

Phase 2: Improvement technique using a hybrid approach employing a randomised iterative improvement algorithm with composite neighbourhood structures and a simulated annealing acceptance criterion.

The pseudo code for the hybrid approach which is implemented in this paper is given in Figure 1. The algorithm starts with a feasible initial solution which is generated by the constructive heuristic in Phase 1 (also as discussed in Abdullah et al. (2005)). In this initial improvement phase presented here, a set of the neighbourhood structures are applied as in Abdullah et al. (2006). The hard constraints are never violated during the timetabling process. Let \( K \) be the total number of neighbourhood structures to be used in the search (\( K \) is set to 11 in this implementation as in Abdullah et al., 2006) and \( f(Sol) \) is the quality measure of the solution \( Sol \). At the start, the best solution, \( Sol_{best} \), is set to be \( Sol \). In a do-while loop each neighbourhood \( i \) where \( i \in \{1, \ldots, K\} \) is applied to \( Sol \) to obtain \( TempSol_i \). The best solution among \( TempSol_i \) is identified, and is set to be the new solution \( Sol^* \). If \( Sol^* \) is better than the best solution in hand \( Sol_{best} \), then \( Sol^* \) is accepted. Otherwise the simulated annealing acceptance criterion is applied. The same parameters as those employed in Abdullah and Burke (2006) are used where the initial temperature \( T_0 \) is equal to 5000, the final temperature \( T_f \) is equal to 0.05 and the number of iterations, \( NumOfIte \) is set to be 200000. At the beginning of the search, \( Temp \) is set to be \( T_0 \). At every iteration, \( Temp \) is decreased by \( \alpha \) where \( \alpha \) is defined as:

\[
\alpha = (\log (T_0) - \log (T_i)) / NumOfIte
\]

This accepts a worse solution with a certain probability. A worse candidate solution is accepted if the randomly generated number is less than \( e^{\delta T} \), where \( \delta = f(Sol^*) - f(Sol) \). Then the current solution is updated. The process continues until the temperature \( T \) is less than the final temperature \( T_f \) or the penalty cost is zero.
Phase 1:
Set the initial solution Sol by employing a constructive heuristic;
Calculate initial cost function f(Sol);
Set best solution Solbest ← Sol;
Set number of iterations, NumOfIte;
Set initial temperature T0;
Set final temperature Tf;
Set decreasing temperature rate as α where α = (log(T0) – log(Tf))/NumOfIte;
Set Temp ← T0;

Phase 2:
do while (Temp > Tf)
    for i = 1 to K where K is the total number of neighbourhood structures
        Apply neighbourhood structure i on Sol, TempSol;
        Calculate cost function f(TempSol);
    end for
    Find the best solution among TempSol, where i ∈ {1, …, K} call new solution Sol*;
    if (f(Sol*) < f(Solbest))
        Sol ← Sol*;
        Solbest ← Sol*;
    else
        Generate a random number called RandomNumber;
        if (RandomNumber ≤ e^{-δ/Temp}) where δ = f(Sol*)-f(Sol)
            Sol ← Sol*;
            Temp = Temp–Temp*α;
        end if
    end if
end while

Phase 3:
Employed a hill climbing method seeded with Solbest as an initial solution;

Fig. 1: The pseudo code for the hybrid approach

Phase 3: Further enhancement using hill climbing method

The hill climbing algorithm starts with an initial solution as provided in Phase 2. (the best solution obtained during phase 2 is seeded as an initial solution). We subsequently chose a course at random and assign it to a randomly available timeslot (the assigned room remains fixed.). The process is repeated until a maximum number of iterations is reached i.e. max_iter, or if there is no improvement on the quality of the solution for a certain number of iterations, num_of_no_improvement. In this experiment, we set the max_iter to be 200,000 and num_of_no_improvement to be 10,000. Though experimentation, it was found that this hill climbing phase does not significantly improve the solution quality (approximately less than 5%) which is produced by the second phase.

4. Experiments and Results

The proposed method was tested on benchmark course timetabling problems from Socha et al. (2002) which are grouped into 5 small (N = 100, R = 5, F = 5 and M = 80), 5 medium (N = 400, R = 10, F = 5 and M = 200) and 1 large (N = 400, R = 10, F = 10 and M = 400) problems. The approach is 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 iterations for our approaches is 200000 (as in Socha et al., 2002) and the best results obtained out of 5 runs are presented.

Table 1 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. (2007), respectively; a variable neighbourhood search (VNS) by Abdullah et al. (2005); and an iterative improvement algorithm with monte carlo by Abdullah et al. (2006). The term “x%Inf” in Table 1 indicates a percentage of runs that failed to obtain feasible solutions.

Table 1: Comparison results on course timetabling problem

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Our approach</th>
<th>Abdullah et al. 2006</th>
<th>Abdullah et al. 2005</th>
</tr>
</thead>
<tbody>
<tr>
<td>small1</td>
<td>0 0 0 0 0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>small2</td>
<td>0 0 0 0 0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>small3</td>
<td>0 0 0 0 0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>small4</td>
<td>0 0 0 0 0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>small5</td>
<td>0 0 0 0 0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>medium1</td>
<td>236 240.2</td>
<td>242 245</td>
<td>338</td>
</tr>
<tr>
<td>medium2</td>
<td>158 160</td>
<td>161 162.6</td>
<td>326</td>
</tr>
<tr>
<td>medium3</td>
<td>261 262.6</td>
<td>265 267.8</td>
<td>384</td>
</tr>
<tr>
<td>medium4</td>
<td>176 178.5</td>
<td>181 183.6</td>
<td>299</td>
</tr>
<tr>
<td>medium5</td>
<td>147 149.5</td>
<td>151 152.6</td>
<td>307</td>
</tr>
<tr>
<td>large</td>
<td>296 315</td>
<td>100%Inf</td>
<td>100%Inf</td>
</tr>
</tbody>
</table>

Table1: Comparison results on course timetabling problem
The best results are presented in bold. In terms of feasibility, we can see that our approach was able to produce feasible solutions for all instances of the datasets. While some 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 and large datasets when compared against all the other methods. For the medium datasets, our approach was able to produce two best results on datasets medium2 and medium5 compared to the other results published in the literature.

We are interested to compare the results from the hybrid approach with our previous results in Abdullah et al. (2006) to show that the simulated annealing acceptance criterion can help to minimise the objective function for this problem compared to the monte carlo acceptance criterion employed in Abdullah et al. (2006). It can be seen that a significantly better result is obtained on the large dataset and slight changes in the penalty cost are observed for the medium datasets.

Figure 2 shows the resulting diagram on large dataset when employing a hybrid approach to course timetabling problems.

### Table 1: Comparison results on course timetabling problem (cont.)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Socha et al. 2002 Ave</th>
<th>Socha et al. 2002 Best</th>
<th>Burke et al. 2003 Best</th>
<th>Burke et al. 2007 Ave</th>
</tr>
</thead>
<tbody>
<tr>
<td>small1</td>
<td>8</td>
<td>1</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>small2</td>
<td>11</td>
<td>3</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>small3</td>
<td>8</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>small4</td>
<td>7</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>small5</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>medium1</td>
<td>199</td>
<td>195</td>
<td>146</td>
<td>372</td>
</tr>
<tr>
<td>medium2</td>
<td>202.5</td>
<td>184</td>
<td>173</td>
<td>419</td>
</tr>
<tr>
<td>medium3</td>
<td>77.5% Inf</td>
<td>248</td>
<td>267</td>
<td>359</td>
</tr>
<tr>
<td>medium4</td>
<td>177.5</td>
<td>164.5</td>
<td>169</td>
<td>348</td>
</tr>
<tr>
<td>medium5</td>
<td>100% Inf</td>
<td>219.5</td>
<td>303</td>
<td>171</td>
</tr>
<tr>
<td>large</td>
<td>100% Inf</td>
<td>851.5</td>
<td>80% Inf 1166</td>
<td>1068</td>
</tr>
</tbody>
</table>

**Fig. 2:** The behaviour of the hybrid approach on the large dataset

The distribution of points in these diagrams shows the correlation between the number of iterations and the overall solution quality. An analysis of the diagrams shows that there is a trend of cost improvement as the number of iterations increases. It shows 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. The slope of the curves indicates a smaller decrease in the penalty cost as the number of iterations increases. Figure 2 also shows that the iterative improvement algorithm with composite neighbourhood structures and the simulated annealing acceptance criterion approach offers more flexibility in accepting a worse solution at the beginning of the search. It can be seen that the points in the graphs are scattered at the early stages of the search but the probability of accepting a worse solution is slowly lowered during the search. We believe that this approach is better than our previous approach in Abdullah et al. (2006) because the monte carlo acceptance criterion is only based on the quality of the solution, however, the simulated annealing acceptance criterion has a cooling schedule that only accept worse solution at the beginning of the search space and towards the end of the search, the probability of accepting worse solution become smaller, which helps to intensify the search to a better local optimal.

### 5. Conclusion

The overall goal of this paper was to investigate a hybrid approach for the course timetabling problem. The hybrid approach is able to obtain best known result on the large dataset. Preliminary comparisons indicate that our hybrid approach works reasonably well across all problem instances and is competitive with other published
approaches. Our future work will attempt to use this technique on real-world course timetabling problems.

References