Using Simulated Annealing to study behaviour of various Exam Timetabling data sets

Edmund Burke* Adam Eckersley* Barry McCollum† Sanja Petrovic*
Rong Qu*

*School of Computer Science & Information Technology
University of Nottingham, Nottingham, NG8 1BB, UK
{ekb,aje,sxp,rxq}@cs.nott.ac.uk

†Queens University of Belfast
Belfast, Northern Ireland
B.McCollum@qub.ac.uk

1 Introduction

In this paper we use a simple simulated annealing (SA) metaheuristic as the basis for comparing behaviour of a number of different examination timetabling problems when optimised from initial solutions given by a largest degree graph-colouring heuristic with backtracking. The ultimate aim of our work is to develop a measure of similarity between exam timetabling problems. By similarity we mean that two *similar* problems will be solved *equally* well and produce results of an acceptable quality using the same meta-heuristic. To help achieve this we are analysing a group of benchmark data sets, all of which have been optimised using SA. We have also developed a Tabu Search algorithm and are working on others to provide comparisons, but for this paper we consider only the behaviour of the data sets when optimised using SA.

This work is motivated by the need for "similarity" measures for large real world problems in order to develop knowledge based systems that can choose the right heuristic to solve given problems [4].

In the following sections we will give a description of the basic exam timetabling problem together with the numerous constraints which may be incorporated and present some of the key features of the data sets studied. We will also give a brief description of how this work should help lead us towards our research goal.

Finally we will present the results we have produced and any conclusions or ideas for further analysis which follow from these results.

2 Exam Timetabling

Examination timetabling is a specific case of the more general timetabling problem. Wren [8] defines the general problem of timetabling as follows:

> "*Timetabling* is the allocation, subject to constraints, of given resources to objects being placed in space time, in such a way as to satisfy as nearly as possible a set of desirable objectives"

In the case of exam timetabling, a set of exams $E = \{e_1, \ldots, e_n\}$ are required to be scheduled within a certain number of periods $P = \{p_1, \ldots, p_m\}$ (which may or may not be fixed beforehand) subject to a variety of hard and soft constraints. Hard constraints must be satisfied in order to produce a feasible timetable, whilst violation of soft constraints should be minimised and provides a measure of how good the solution is via the objective function [1].

The main hard constraints in exam timetabling are usually represented by the following:

- Every exam in the set E must be assigned to exactly one period, p, of the timetable.
- No individual should be timetabled to be in two different places at once, i.e. any two exams, e_i and e_j which have students in common must not both be scheduled in the same period, p.
- There must be sufficient resources available in each period, p, for all the exams timetabled, e.g. room capacities must not be violated.

Individual institutions may also have their own specialised hard constraints based on the needs of their courses and resources [1]. Any timetable which fails to satisfy all these constraints is deemed to be infeasible.

Soft constraints are generally more numerous and varied and are far more dependent on the needs of the individual problem than the more obvious hard constraints. It is the soft constraints which effectively define how good a given feasible solution is so that different solutions can be compared and improved using an objective function. Burke and Petrovic [3] list the more common soft constraints as being:

- *Time assignment* - An exam may need to be scheduled in a specific period
- *Time constraints between events* - One exam may need to be scheduled before, after or at the same time as another
- *Spreading events over time* - Students should not have exams in consecutive periods or two exams within x periods of each other
- *Resource assignment* - An exam must be scheduled into a specific room

Exam timetabling problems are often modelled as graph colouring problems with nodes representing the exams and the edges representing clashes between exams [2]. Edges may have weights to represent the number of students involved in a given clash. With this representation, a complete k-colouring of the graph with no connected nodes having the same colour can be mapped directly onto an exam timetable solution with each colour representing a period of the timetable. Graph-colouring algorithms can often be used to provide good initial solutions for improvement using meta-heuristics.

3 Benchmark Data Sets

Heuristics are often developed to solve a specific problem at a given institution. In many cases, these heuristics may not provide good results when applied to a different problem which has a very different structure and set of constraints. The aim of our research is to develop a measure of similarity between exam timetabling problems where the similarity measures the suitability of heuristics for those problems. For example, if Tabu Search works well on 3 different problems they would be deemed to be similar. In this paper we use a simulated annealing algorithm to produce sets of results for a number of benchmark problems and examine the behaviour of each problem with respect to this heuristic and attempt to draw some conclusions regarding key aspects of similarity between problems.

Whilst many heuristics are developed specifically to solve the timetabling problem at a given institution [6], there are a number of benchmark data sets available for comparing the performance of different methods also. The most widely used of these are the Carter benchmarks [7]. These have no side-constraints other than the main soft-constraint of spreading clashing exams throughout the timetable as far as possible from each other, but provide a good measure of the ability of a heuristic to solve the core problem with the key hard constraints. The objective function used by Carter to measure the quality of a solution is based on the sum of proximity costs given as:

$$w_s := \frac{32}{2^s},\ s \in \{1, \ldots, 5\}$$

where w_s is the weight given to clashing exams scheduled s periods apart.

This weighting is then multipled by the number of students involved in the clash. In essence, this means that a bias is given to clashing exams with more students in common so that these are spread further apart. Clashing exams involving fewer students will give a lower overall penalty for a given number of periods apart and are therefore regarded as less important. In this paper we also present results for these same data sets using a simplified objective function which ignores the number of students involved in each clash and therefore weights all clashes equally, whether they have just 1 student in common or 100. Both of these objective functions can be viewed as being worthwhile from different perspectives and give rise to a number of differences in how problems can be regarded as similar or not. The main defining features of the data sets used are given in Table 1.

For the purpose of measuring similarity between problems, many other features have to be considered which will have a larger impact on the similarity of two problems. Most of

Table 1: Features for Carter Data Sets

Data Set	No. of exams	No. of students	No. of enrollments	Conflict Matrix Density	No. of periods
CAR-S-91	682	16925	56877	0.13	35
CAR-F-92	543	18419	55522	0.14	32
EAR-F-83	181	1125	8109	0.27	24
HEC-S-92	81	2823	10632	0.20	18
KFU-S-93	486	5349	25113	0.06	20
LSE-F-91	381	2726	10918	0.06	18
STA-F-83	139	611	5751	0.14	13
TRE-S-92	261	4360	14901	0.18	23
UTA-S-92	622	21267	58979	0.13	35
UTE-S-92	184	2750	11793	0.08	10
YOR-F-83	190	941	6034	0.29	21

these will come from a statistical analysis of the datasets and their behaviour when various meta-heuristics are applied to them. In the following section we present results of simulated annealing applied to these data sets using both objective functions mentioned above and aim to draw some conclusions regarding how consistent the algorithm is over 10 runs on each problem, how large an impact the initial solution has on the best solution found and how the different objective functions affect this behaviour.

4 Results & Points to note

The results presented in this section are taken from 10 runs of our simulated annealing (SA) algorithm on each data set and for each optimisation function. The initial solution to be fed to the SA heuristic is given by a largest degree graph colouring heuristic with backtracking which gives a feasible solution. Our SA heuristic selects moves from a neighbourhood defined by moving an exam, e, from one period, p_1 to another, p_2. The exam and period are both chosen at random and only moves leading to a feasible solution are allowed. The chosen feasible move is accepted or rejected using the standard probabilistic acceptance criteria of SA with improving moves always accepted and moves leading to an inferior solution accepted with decreasing probabilty based on a geometric cooling schedule. The starting temperature and cooling schedule were selected at random and tuned based on the results produced. A number of improvements can be made to improve the perfomance of our SA heuristic, but for our purposes the results produced are of an acceptable standard for comparing similarity between problems.

In Tables 2 & 3, the same initial solution is used for all 20 runs of the SA heuristic for each data set. The first 10 runs (Table 2) were optimised using the original Carter's function including student weights for clashes (the Weighted Set), whilst the second set of 10 runs (Table 3) was optimised using the simplified function weighting all clashes based purely on distance apart in the timetable. The values given for Initial Solution in Tables 2 & 3 for a

Table 2: Results from our Simulated Annealing heuristic initialised by the same solution each time, using the standard Carter's optimisation function

Data Set	Initial Solution	Percentage Improvement	Standard Deviation	Average Final Solution
CAR-S-91	160224	22.2%	1.02%	124648
CAR-F-92	141506	24.8%	1.43%	106367
EAR-F-83	63933	11.1%	1.49%	56846
HEC-S-92	53598	22.9%	1.81%	41308
KFU-S-93	190237	47.0%	2.22%	100790
LSE-F-91	56393	20.0%	0.95%	45125
STA-F-83	101100	3.4%	0.68%	97701
TRE-S-92	59256	15.7%	1.54%	49944
UTA-S-92	141809	18.3%	1.02%	115827
UTE-S-92	116735	31.8%	2.44%	79587
YOR-F-83	49698	9.3%	1.04%	45073

given data set represent the exact same solution, but measured by the two different functions.

Tables 4 & 5 show results when a different initial solution is fed to the SA heuristic for each of the 20 runs across the two objective functions. Table 4 using the Carter's function, Table 5 using the simplified function. For these experiments, a number of other statistics are also noted in addition to those also used in Tables 2 & 3.

Due to space constraints, some words in Tables 4 & 5 were abbreviated as follows: Avg. = Average, Init. = Initial, Soln. = Solution, Imp. = Improvement, Std Dev = Standard Deviation.

Of course, 10 runs for each data set and optimisation function is not statistically representative. However, in this phase of the research work we aim to provide a guideline for further research and intend to perform more runs in further experiments. The obtained results do provide some interesting points for deeper study since the standard deviations will still be of the same order for a larger number of runs. The results presented for Standard Deviation are calculated as a percentage of the average for a given set of results.

It is also worth noting that the initial solution used for each data set in the first set of experiments may be either good or bad so any absolute analysis of the percentage improvement from this solution is not worthwhile. The second set of experiments from 10 different initial solutions each time aims to overcome any bias from a bad or good initial solution.

From the results presented in Tables 2 & 3, it can be seen that in all but one case, the SA heuristic improves from the same initial solution 10 times to within a standard deviation of just 3% from the average, indicating that the data sets are relatively stable with respect to our SA heuristic, i.e. when seeded with the same initial solution, the results produced are consistent over 10 runs without huge variations. The one exception to this is the STA-F-83 data set when optimised using the simplified function eliminating student weightings which gives a standard deviation of $> 6\%$. Referring to Table 5, we can see that this behaviour is still apparent

Table 3: Results from our Simulated Annealing heuristic initialised by the same solution each time, using the simplified optimisation function without student weights

Data Set	Initial Solution	Percentage Improvement	Standard Deviation	Average Final Solution
CAR-S-91	51091	20.0%	0.70%	40873
CAR-F-92	38634	21.6%	0.90%	30289
EAR-F-83	11316	12.3%	1.20%	9924
HEC-S-92	4329	6.5%	1.10%	4048
KFU-S-93	17616	23.8%	0.70%	13423
LSE-F-91	14613	29.5%	1.00%	10302
STA-F-83	6193	25.3%	*6.80*%	4626
TRE-S-92	16194	19.9%	1.30%	12971
UTA-S-92	41331	25.2%	0.90%	30916
UTE-S-92	8366	22.0%	2.60%	6525
YOR-F-83	13392	10.7%	1.20%	11959

Table 4: Results from our Simulated Annealing heuristic initialised by a different solution each time, using the standard Carter's optimisation function

Data Set	Avg. Init. Soln.	Std Dev	% Imp.	Std Dev	Avg. Final Soln.	Std Dev	Best Imp.	Best Solution	% imp. for Best Solution
CAR-S-91	152278	3.33%	24.1%	2.22%	115633	3.94%	27.8%	110476	27.8%
CAR-F-92	141332	2.95%	23.5%	3.66%	108023	3.09%	27.8%	101866	25.9%
EAR-F-83	65586	4.57%	15.0%	3.32%	55676	3.58%	20.0%	52827	13.9%
HEC-S-92	55412	*8.27*%	20.5%	*10.13*%	43730	3.21%	33.3%	40413	31.6%
KFU-S-93	161601	*10.77*%	36.1%	*11.85*%	102115	3.89%	47.2%	98688	40.6%
LSE-F-91	57986	3.60%	25.2%	4.31%	43362	4.87%	30.5%	37764	30.5%
STA-F-83	106717	2.89%	6.40%	1.45%	96564	2.71%	8.9%	96564	5.1%
TRE-S-92	58615	3.95%	13.1%	1.71%	50953	3.77%	16.3%	48234	12.8%
UTA-S-92	128946	4.06%	20.9%	2.60%	102074	5.61%	24.5%	90914	24.5%
UTE-S-92	116636	*10.50*%	24.1%	*6.29*%	82680	4.64%	31.3%	82680	19.7%
YOR-F-83	50142	2.03%	12.0%	2.98%	44109	3.12%	16.6%	42001	16.6%

Table 5: Results from our Simulated Annealing heuristic initialised by a different solution each time, using the simplified optimisation function without student weights

Data Set	Avg. Init. Soln.	Std Dev	% Imp.	Std Dev	Avg. Final Soln.	Std Dev	Best Imp.	Best Solution	% imp. for Best Solution
CAR-S-91	51589	1.30%	20.7%	1.25%	40908	1.23%	22.0%	39994	22.0%
CAR-F-92	38339	1.32%	21.5%	0.93%	30094	1.21%	22.7%	29616	21.4%
EAR-F-83	11794	2.21%	14.5%	2.20%	9938	1.32%	17.5%	9938	14.9%
HEC-S-92	4425	3.19%	7.0%	3.34%	4114	2.94%	12.6%	3926	9.8%
KFU-S-93	17355	1.98%	25.4%	1.62%	12939	1.63%	27.1%	12590	27.1%
LSE-F-91	14959	2.02%	30.9%	2.30%	10332	1.50%	34.5%	10034	34.5%
STA-F-83	6025	4.18%	18.3%	*8.55%*	4917	*8.95%*	29.3%	4167	29.3%
TRE-S-92	15749	1.87%	18.9%	3.02%	12775	1.73%	23.8%	12335	23.8%
UTA-S-92	41770	1.25%	25.4%	1.81%	31145	0.90%	28.3%	30542	28.3%
UTE-S-92	8039	5.19%	25.4%	5.08%	5997	*7.02%*	33.0%	5253	30.6%
YOR-F-83	13270	1.80%	10.8%	2.68%	11832	1.92%	14.4%	11338	14.4%

when 10 different initial solutions are used, whereas with the Carter's optimisation function, Table 4 confirms that this same data set is at least as stable as the rest. This indicates that the function used to optimise the data set (i.e. the definition of what a good timetable is) can have a major effect on the consistency of results produced by the algorithm, indicating that this particular data set is similar in behaviour to others when using one function, but very different when using a different function. It can also be noted from Table 5 that this data set shows very different behaviour from the rest when considering the variation of initial solutions vs final solutions. In most cases the standard deviation of the initial solutions and final solutions are fairly similar, yet for the STA-F-83 data set, the SA heuristic introduces a large amount of further variation into the final solution than was present in the 10 initial solutions. From Table 4, we can see that when a variety of initial solutions are used to seed the SA heuristic, the HEC-S-92 and KFU-S-93 data sets suddenly start to produce a much wider range of final solutions with standard deviations of the order of 10%. In the case of the KFU-S-93 data set, it can be seen that the initial solution used in Table 2 was very bad relative to the average initial solution used in Table 4. This, together with the high deviation in initial solutions for this data set helps to explain why the % improvments are so varied. Likewise the HEC-S-92 and UTE-S-92 data sets show a wider range of initial solutions leading to a wider range of % improvements. Despite this though, the Average Final Solutions produced in Table 4 show similar deviations over all data sets. This indicates that although the initial solutions for some data sets show a higher variation, the SA heuristic flattens this out when producing the final solutions by improving the worse initial solutions notably more than the better ones.

It is also interesting to note from Tables 4 & 5 that the best final solution and the best percentage improvement for a given data set come from the same run (initial solution) in a large number of cases[1]. Further research with more runs would need to be done before any

[1]Each run takes between 10 seconds and 3 minutes depending on the data set

conclusions can be drawn from this, but it does indicate that in many circumstances, the best solution that the algorithm can find comes from a large improvement from the initial solution rather than from the best initial solution. Analysis of the individual runs for these data sets does indeed show that in many cases some of the best final solutions come from the worst initial solutions. For other data sets though this is markedly not the case and the better final solutions generally come from the better initial solutions indicating a much stronger dependence on having a good initial solution for these data sets in order for the SA heuristic to perform well. In the case of the LSE-F-91 data set in Table 4, the biggest improvement from an initial solution actually comes from the best initial solution yielding a final solution far better than in any of the other 9 runs. This may just be an anomoly of the small number of runs used, but is certainly worthy of further investigation.

Finally, when comparing results across the two objective functions it can be seen that there are some very striking differences between certain data sets. Most notably, the HEC-S-92 and KFU-S-93 data sets are improved a great deal more from all the initial solutions when optimised using the standard Carter function than when optimised using the simplified function. On the other hand, the LSE-F-91, STA-F-83 and TRE-S-92 data sets show completely the opposite behaviour and are improved more from an initial solution using the simplified objective function than the standard Carter's objective function. This again indicates that when measuring similarity between two data sets, the objective function used has a major impact.

5 Conclusions

To conclude, it can be seen from our results that there are a number of interesting differences in the behaviour of some of the data sets when compared against each other for the same objective function and also when compared with the same data set optimised using a different objective function. Since these results are all produced from sets of just 10 runs, further tests will need to be done with many more runs in order to come to any solid conclusions, but our initial results indicate many areas in which this further analysis can be done and provide very worthwhile results. These include further runs of the heuristic on the data sets which provide very varied results (high standard deviation) to see if this is still the same over 100 or more runs, a deeper analysis of the STA-F-83, HEC-S-92, UTE-S-92 and KFU-S-93 data sets to study the large variation in these results when run from 10 different initial solutions and a further examination into exactly which data sets rely strongly on their initial solutions and which produce equivalent quality solutions irrespective of the initial solution.

Measuring similarity between two complex optimisation problems is far from an exact science and it will never be possible to say with absolute certainty that two problems are similar and that the heuristic which performs best on one will also perform best on the other, but through our analysis we hope to gain a better understanding of the key factors affecting how similar data sets are with respect to the behaviour of heuristics applied to them.

References

[1] Burke, E.K., Elliman, D.G., Ford, P., Weare, R.F. *Examination Timetabling in British Universities - A Survey*, in [5]

[2] Burke, E.K, Elliman, D.G. and Weare, R.F. *A University Timetabling System based on Graph Colouring and Constraint Manipulation* Journal of Research on Computing in Education Volume 27 Issue 1 Fall 1994, pages 1-18

[3] Burke, E.K. and Petrovic, S. *Recent Research Directions in Automated Timetabling*. European Journal of Operational Research - EJOR, 140/2, 2002, 266-280

[4] Burke, E.K., MacCarthy, B., Petrovic, S. and Qu, R. *Structured Cases in CBR: Re-using and Adapting Cases for Timetabling Problems*. Knowledge-Based Systems 13, pp. 159-165, 2000

[5] Burke, E.K. and Ross, P., editors. *Practice and Theory of Automated Timetabling*, volume 1153 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, Heidelberg, 1996.

[6] Carter, M. W. and Laporte, G. *Recent Developments in Practical Examination Timetabling* In [5].

[7] Carter, M. W., Laporte, G. and Lee, S. Y. *Examination timetabling: Algorithmic strategies and applications*. Journal of Operational Research Society, 74:373-383, 1996.

[8] Wren, A. *Scheduling, timetabling and rostering - A special relationship?*. In [5], pp. 46-75.