University exam scheduling using a two stage genetic algorithm

Tamás D. Gedeon and Nicholas Jenkins
Department of Information Engineering
School of Computer Science & Engineering
The University of New South Wales
Sydney NSW 2052 AUSTRALIA
E-mail: tom@cse.unsw.edu.au

Abstract — A method of scheduling complex examination timetables, such as found at most universities, is presented using genetic algorithms. This extends previous work by splitting the problem into sequential components, two of which are solved using genetic algorithms. Also, it can be seen that some subjects are likely to be harder to schedule correctly than others, so a method is described of linking the reward for the clash-free scheduling of these subjects to scheduling difficulty. Finally, in situations where a clash-free timetable cannot be found, it is very desirable to minimise the number of students who experience clashes.

I. INTRODUCTION

Scheduling examinations for large student bodies is a real and recurring problem faced by many secondary and tertiary academic institutions. The simplest form of the problem is to allocate students, invigilators (exam supervisors) and examinations to rooms and times in such as way that no student is scheduled to be sitting more than one exam at any given time. Increasing enrolments worsens the timetabling problem, as do moves towards more modularized degrees [1], as both increase the density of the problem – the probability that any two given subjects will have some students in common. One approach that can be used in scheduling examinations is genetic algorithms.

Genetic Algorithms (GAs) are a class of heuristic search algorithms. The natural analogy is from population genetics, whereby the Darwinian principal of survival of the fittest is applied to individuals over time. Some individuals are deemed to be fitter than others, because more of their offspring survive. This process involves offspring taking portions of DNA from both of its parents.

When transferring this analogy to a computer algorithm we can apply GAs to combinatorial optimisation problems that we wish to solve. Most crucially though, we must decide how we are going to encode a solution to that problem. This solution encoding is called a chromosome, and is analogous to DNA. This chromosome can be represented as a bit string of fixed or variable length. A fitness function, is then created that is intended to reward good attributes of a chromosome and punish bad attributes by adding to or deducting from the total fitness of that solution. If the fitness function is well designed, and a good representation has been chosen, then a GA will usually find good solutions given sufficient processing time, but with increasing problem size, finding a workable solution in a reasonable time can be a problem.

II. EXAM SCHEDULING

Timetabling exams is the process of assigning exams to rooms and timeslots. It is a multi-constrained combinatorial optimisation problem that is known to be NP-hard [3]. To the basic constraints of the timetabling problem (eg. no clashes, room capacity not exceeded), constraints of varying importance can be added dependant upon the requirements of the institution. An example is large subjects to be timetabled earlier to allow the markers more time to assess the larger number of papers.

As the GA is designed to be applied to determining an end of semester examination timetable at the University of New South Wales (UNSW), in Sydney, Australia, some assumptions from this environment were made. It is assumed that there are two examination periods per day (morning and afternoon), that multiple exams of different length can be assigned to the same room, and that no exam will take longer than one examination period (which in practice means a maximum of three hours).

III. PREVIOUS WORK

Previous work in this area include studies by Corne *et al.* [1] at the University of Edinburgh, UK and by Burke *et al.* [2] at the University of Nottingham, UK. Corne *et al.* created a system that is used for scheduling times for the Master of Science exams at the University of Edinburgh, but did not apply the GA to scheduling rooms. Burke *et al.* used a direct representation that specified the time and location of an exam, but only considered exam spacing issues for adjacent exams. We have also done some previous work using GAs [3] at UNSW.

IV. FOUR STAGE PROBLEM STRATEGY

We decomposed the problem of producing a full exam schedule to the following 4 sequential stages:

A) Best-Groupings;
B) Smooth;
C. Timeslot-Assignment; and
D. Room-Assignment.

A Best-Groupings

Carter *et al.* [7] suggest first finding the "maximum clique," being a subset of exams, each of which must be scheduled into a different timeslot, or a clash will occur. By finding the largest maximum clique in a set of exams we accomplish several things:

- A maximum clique will often contain some of the hardest to schedule exams, and removing these from the pool of exams to be scheduled simplifies the remaining problem.

- Removing the maximum clique exams (irrespective of their difficulty to schedule) from the pool of unscheduled exams helps reduce the size and hence the computational complexity of the problem.
- The size of the clique provides an absolute lower bound on the number of timeslots for a clash-free timetable.

The algorithm we used for finding the maximal clique is called DFMAX [8]. The members of the maximal clique are assigned timeslots and removed from the pool. The possible choices of timeslots irregardless of clashes for the remaining exams can be represented as a tree structure. We then perform a recursive depth first tree traversal. This is an exact algorithm, which for an NP-hard problem may not terminate in a reasonable time, hence we added some heuristic optimisations.

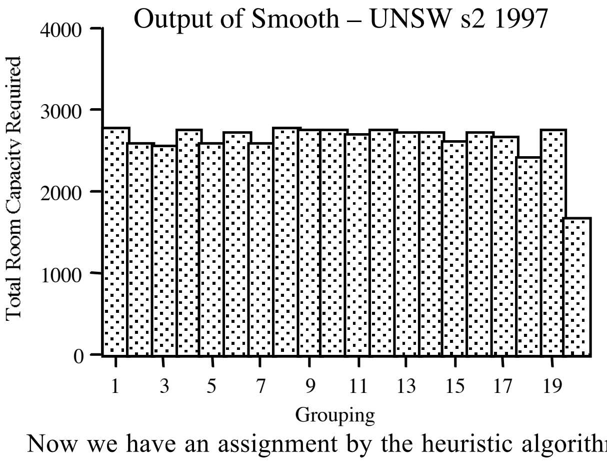
Firstly, a maximum time to spend to find a solution using a certain number of timeslots is imposed. If the time is exceeded the algorithm restarts (with one more timeslot) proceeding with the best cached schedule (most exams scheduled without clashes). Secondly, the exam to insert into the schedule is the one which has the highest ratio of number of clashes if scheduled over number of timeslots the exam can go into.

B Smooth

Once we have produced a schedule as above, the problem we now face is that the room capacity is likely to be exceeded at some point. Note that determining a good concurrent grouping of exams to avoid clashes, and finding groupings which fit into the toral room capacity are completely different problems, so it is unlikely that a good solution to the first will also be a good solution to the second.

The heuristic smoothing algorithm is very simple. An exam is selected and moved to a random other position if this can be done without creating clashes. This is done repeatedly for a random exam and then an exam with large room capacity requirements. The minimum of the largest requirement for room capacity found is cached.

C Timeslot-Assignment



Now we have an assignment by the heuristic algorithms of exams to clash-free groupings, with a reduced total room capacity requirement. The final two steps are to map these grouping to actual timeslots, and to assign these exams to rooms. These are largely separate problems, so there is some choice in which order to solve these problems. We elected to solve the mapping first and then the room allocation (next section) second for two reasons:

1) Not all rooms may be available in all timeslots. As such it seems logical to determine which timeslot a particular set of exams will be in, and then fit these exams to the available rooms.
2) At UNSW, the first schedule published (the preliminary timetable) lists only the timeslots for each exam, but not the room for each exam. It would seem reasonable to structure, where possible, the method we use to produce results in the order that we require them.

The GA step of mapping groupings to timeslots is concerned principally with satisfying:

1) Schedule larger exams earlier – there should be no large exams towards the end of the schedule.
2) Find a good spacing of exams for students.
3) Attempt to satisfy any preferences for a particular timeslot for a particular exam.
4) Not placing postgraduate exams too close to the end of the schedule.

Of these constraints, the first two were considered by the examinations section to be the most, so they were weighted to account for the largest components of the fitness function. The best solution found by this GA is below.

Constraint	value	weight	Result
Groups with largest subject sizes earlier, smaller later	108	80	8660
Spacing of exams for students	779587	0.012	9355
Preferences for Particular dates (*eg* Chemistry & Maths)	7	160	1120
Penalty for Scheduling Postgrad Exams towards the end	10	300	3000
Fitness Function Total:			22135

To determine the mapping from groupings to timeslots, a concise direct representation was used, such that each group mapped to a unique timeslot.

It is worth elaborating how penalties for each preference is calculated. For the preference that large exams be placed early in the schedule, the size of the largest exam in each grouping is determined. A penalty is added if the size of this exam exceeds a value depending on its timeslot: earlier timeslots have a large value (>1000 students), while the last few timeslots have a low value (<150 students).

Determining the exam spacing uses the number of common students and a multiplier dependant on the number of gaps (possible timeslots) between the two timeslots, with 7 or more gaps giving a zero multiplier. We count virtual timeslots for each evening, and morning and afternoon timeslots on weekends, though no actual exams are scheduled. Thus, two exams the same day have no gaps, two morning exams on subsequent days have 2 gaps for the afternoon real timeslot and the evening virtual timeslot, and a Friday afternoon exam followed by a Monday morning exam has 7 gaps being the Friday evening and three gaps each for Saturday and Sunday being the morning afternoon and evening times.

The penalty value for "preferences for particular dates" was determined by counting the number of timeslots distance from the preferred timeslot (if this was specified). The penalty for postgraduate exams was the number of postgraduate exams held in timeslot 18 and later, as the examinations section indicated that it would be preferable to schedule most postgraduate exams earlier.

The settings we used were to terminate when no improvement in the fitness of the best solution has been made for 3000 , a crossover rate of 0.5, and a mutation rate of 0.1, and a population size of 40.

D Room-Assignment

The final stage of our approach is to allocate exams to one or more rooms, by trying to solve as much of the problem as possible allocating each exam to only one room, then 'carrying' any excess students over to the second room, and so on.

The primary concern in this step was to minimise the amount of exceeded room capacity. However, it was not possible to eliminate instances of exceeded room capacity, as several of the largest exams are have more students than the capacity of the largest room. We hope to minimise split exams where students have to be carried over into another room. One point to note when carrying students forward into the next room is that the number of students in any room for a split exam must be greater than or equal to 20 students to ensure exams are not split in an absurd manner, with students from an exam scattered in ones and twos.

We also want to minimise the number of mixed duration exams, when exams in a room in a timeslot are not all of the same length. The effect is to act as a major disturbance on the remaining students when the earlier exam leaves.

A less important concern is to reduce the number of times an exam room is reopened. An exam room is reopened when it is used in a timeslot after not having been used in the previous timeslot in the schedule. The invigilators are assigned to one room for the duration of the schedule, and how consistent their schedules are can be determined from the number of times a room is reopened.

The chromosome used in this GA had a structure as shown below (note that only the first 7 subjects of 677 are shown). The method we used to generate rooms is the 'recipe' method, where the set of variable that are to be adjusted can be varied independently of one another, and we set the range of integers, where each integer corresponded to a different room.

Exam	Students	1st Room	Students	2nd Room
ACCT1501S2	250	23		
ACCT1511S2	936	16		
ACCT2542S2	482	23		
ACCT2552S2	110	17		
ACCT3563S2	57	26		
ACCT3583S2	448	18		
ACCT3593S2	21	5		

After the first step is completed, the students who have not been allocated are carried over into the next room.

The settings used for the room allocation GA were to terminate after 30,000 trials, with a crossover rate of 0.5, a mutation rate of 0.015, and a population size of 20.

A very low mutation rate was required because of the large size of the chromosome being used. Setting the mutation to above 0.03 produced a sharp decline in the speed at which results were produced. The best solution found by this GA is below.

Constraint	value	weight	Result
Exceeding room capacity	14862	1	14862
Instances of exceeded capacity	174	50	8700
Wasted room capacity	8862	0.5	4431
Mixed duration penalty	3	200	600
Reopening an exam room	96	30	2880
Total Fitness			31473

Instances of exceeded capacity is penalised as well as total exceeded capacity to reduce the number of exams split, which is different to the number of students encountering split exams.

V. MODIFIED HYBRID STRATEGY

The room allocation genetic algorithm performance was not satisfactory in results or the very long times taken to produce results due no doubt to the very long chromosome used, so a heuristic algorithm was written:

1) Allocate exams which require specialised rooms first, starting with the most restrictive. So, allocate drawing then openbook, and finally normal exams.
2) For each category allocate the largest exams first.
3) Once a room has been allocated an exam, only exams of the same duration will be allocated to it.
4) Initial allocation of an exam to a room ignores the number of spare seats in the room.
5) All other suitable rooms are examined, and the exam is moved if: i) the unused capacity in another room is larger and that extra space is required for this exam; or ii) if the unused capacity is less but this exam would fit into it.
6) Any exams exceeding room size are split, with at least 20 students in each additional room.

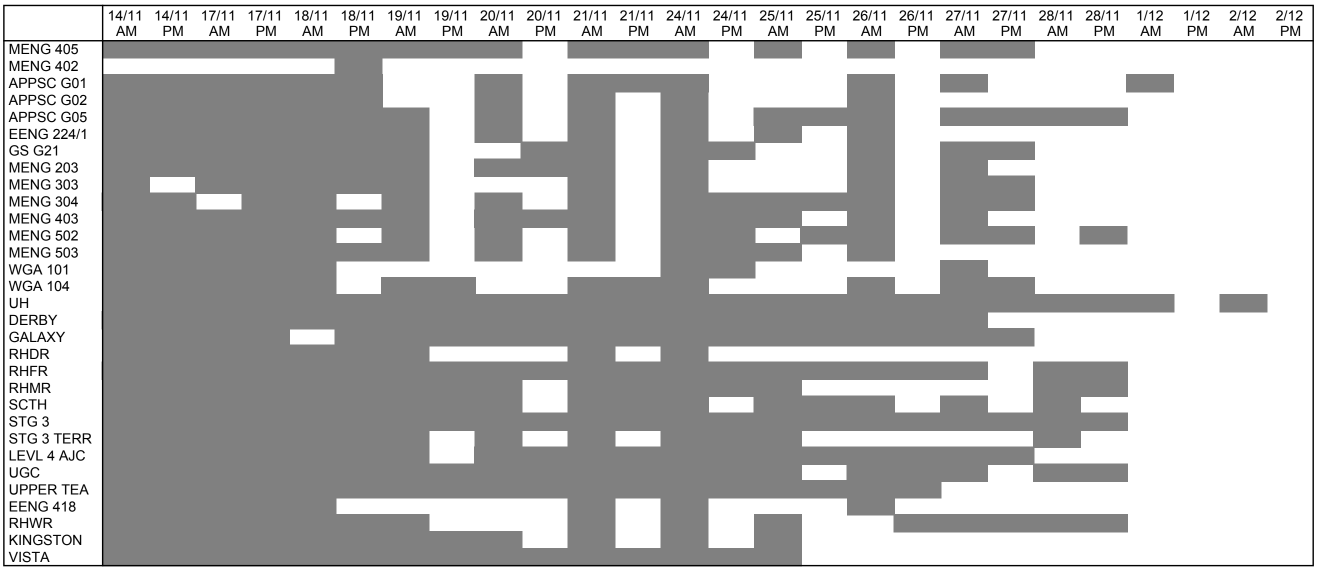
The above heuristic successfully allocates 99% of the exams in the UNSW session 2 1997 exam schedule. There are two special cases introduced to deal with the large drop in capacity from the largest to the next largest room, from 745 for "UniSearch House" to 245 for "Stage 3".

7) Only allocate exams of 180 or 120 minutes to "UniSearch House". That is, allow no short or unusual length exams there.
8) Allocate any other exams to "UniSearch House" if they have the same duration as the other exams there, have 50 or more students and will still fit in there.

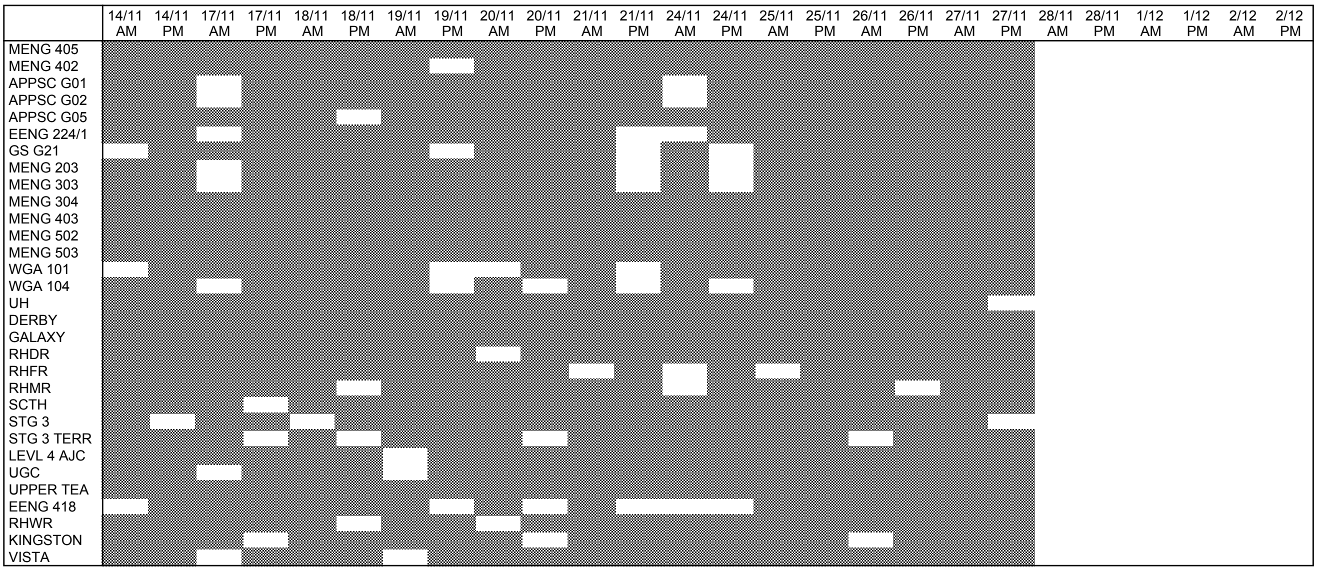
The above heuristic now allocates 100% of the exams in the UNSW session 2 1997 exam schedule.

VI. RESULTS

A UNSW Final Schedule (Exams branch)



B UNSW Final Schedule (Our result)



Our hybrid genetic algorithm / heuristic technique has produced a shorter exam schedule which is more densely packed, as can be seen from the room allocation diagrams shown above. Our technique requires more rooms, as we used only the lower room sizes of multiple use rooms (a room suitable for open book exams could seat more if used in a normal examination mode). The number of timeslots required using our hybrid GA final schedule was 20 and required 2752 maximum total room capacity, whereas the UNSW final schedule required 24 and 3286 seats.

C Map Groupings to Timeslots

The fitness of the best solution found was shown earlier, taking 19610 trials, which took approximately 6 hours on a high powered PC. The small size of the chromosome meant that evaluating each solution was quite fast – the GA evaluated 3,400 trials per hour.

D Largest Exam Size in Each Timeslot

In comparison between the UNSW final schedule and our schedule, the first half is fairly similar for both schedules, but after our hybrid GA schedule is better, with the largest exam in each slot often being 100 students smaller than for the UNSW final schedule.

E Postgraduate Exams

The number of postgraduate exams after the 26th November was 10 for the hybrid GA final schedule compared to 7 for the UNSW final schedule.

F Preferred Timeslots

The preferred timeslots for First Year exams were accommodated in the UNSW final schedule, whereas for the hybrid GA final schedule, only half were in the preferred timeslots. The other half were less than 4 timeslots away.

G Exam Spacing

The UNSW final schedule produces fewer gaps between exams for all size gaps except for size two than our hybrid GA final schedule. This means our schedule produces more exams with no gaps between exams and only 1 gap, but provides more larger gaps. We suspect that students are more likely to complain about no gaps in their schedule, than be pleased about large gaps.

H Allocate Rooms to Exams: GA versus heuristic

Using two rooms the GA room allocation was unable to allocate a room for 11% of student exams. The GA was terminated at this point, as an analysis of the partial results showed that in a number of cases the GA either could not to schedule the remaining exams to the correct type of room, or would not be able to schedule the remaining exams without creating an unacceptably high level of split exams and mixed duration exams. An analysis of this partial room allocation revealed 150 split exams, 63 reopened rooms, and 16 instances of mixed duration. The time taken for this partial room allocation was 72 hours.

The diagram on the previous page shows the room use produced by the heuristic room allocation. Note that this heuristic was able to schedule every exam fully to the correct room type. The time taken to produce this result was 8 seconds.

I Split exams

Our hybrid GA final schedule was able to produce fewer split exams (86) than the UNSW final schedule (94), a decrease by 9 %. Also the total number of times any exams were split by the hybrid GA final schedule was fewer (108 compared to 125), a decrease by 14 %.

J Mixed Duration

The number of incidences of mixed duration was 17 in the UNSW final schedule whereas there were no incidences in the hybrid GA final schedule.

K Reopened Rooms

The number of times a room was reopened in the UNSW final schedule was 85, whereas our schedule only reopened 54 times (a decrease of 37 %).

VII. CONCLUSION

We have shown that it is possible to produce good examination schedules using a hybrid genetic algorithm and heuristic strategy. In the process we come to some conclusions as to the suitabilities of genetic algorithms.

GAs are best suited to certain situations. One of these is when studying certain types of non-trivial problems, and looking to produce reasonable results quickly. A second situation is one where a specialised approximation algorithm is infeasible, possibly because the constraints are mutually competitive, or because the significance or weighting attached to each preference can vary over time. An example of this type of situation is the second step of the exam scheduling process, where the assignment of groups to timeslots is performed so as to meet as many competing preferences as possible.

Our results for the room allocation GA were extremely poor, in both in terms of the quality of result, and the time taken to achieve this result. Whilst it is true that the exam scheduling problem is not a time-critical problem, any process that takes 3 entire days to produce only partial results seems unacceptable. On the other hand, an overnight process probably is acceptable, because the machine that generates the schedule can then be used normally during the day.

These results illustrate the areas where GAs are unsuitable: namely where there is a good heuristic algorithm, or where the chromosome's size is excessively large. In both of these situations, a GA is likely to take a long period of time to produce substandard results. Problems like the mapping of groupings to timeslots however, where the chromosome size is small and scales well with problem size, and where a good heuristic is likely to be hard to create, and where a brute force approach is inappropriate, are likely to produce reasonable results using GAs.

VIII. REFERENCES

[1] D. Corne, H.-L. Fang and C. Mellish, “Solving the modular exam scheduling problem with genetic algorithms,” *Proceedings 6th Int. Conf. on Industrial and Engineering Applications of AI and Expert Systems*, Chung, Lovegrove and Ali (eds.), Gordon & Breach, Langhorne, Pa. , 1993, pp 370-373.

[2] E.K. Burke, D.G. Elliman and R. Weare, “A genetic algorithm based university timetabling system,” *Proceedings 2nd East-West Int. Conf. on Computers in Educucation*, Ukraine, 1994, vol 1, pp 35-40.

[3] Jenkins, N. and Gedeon, T.D., “Genetic algorithms applied to university exam scheduling,” *Proceeding of Int. Conf. on Neural Information Processing*, 1997, pp. 1034-1037, Dunedin, 24-28 Nov 1997. also is Kasabov, N, O’Shea, R, Coghill, G and Gedeon, TD, (eds.) *Progress in Connectionist Based Information Systems*, Springer-Verlag, Singapore, 2 vols., 1997.

[4] S. Even, A. Itai and A. Shamir, “On the complexity of timetable and multicommodity flow problems,” *SIAM Journal of Computing*, vol. 5, no. 4, 1976, pp. 691-703.

[5] R. Nakano and T. Yamada, “Conventional genetic algorithm for job shop problems,” *Proceeding 4th Int. Conf. on Genetic Algorithms*, Los Angeles, 1991.

[6] N. Jenkins, “A hybrid approach to genetic algorithms: Scheduling exams at the University of New South Wales,” Honours Thesis, Sydney, 1997.

[7] M. Carter, G., Laporte and S. Lee, “Examination timetabling: algorithmic strategies and applications,” *Journal of the Operational Research Society*, vol. 47, 1996, pp. 373-383.

[8] D.S. Johnson, “DFMAX source code,” AT&T Bell Labs, ftp://dimacs.rutgers.edu/pub/dsj/clique, 1993.