1999 Third International Conference on Knowledge-Based Intelligent Information Engineering Systems, 31st Aug-1st Sept 1999, Adelaide, Australia

Genetic Algorithms vs. Tabu Search in Timetable Scheduling

S. C. Chu*†
†National Kaohsiung Institute of Technology, Taiwan, R.O.C.
*University of South Australia, Australia

H. L. Fang
AI Application Group
E. & C. Dept, China Steel Corp.
Taiwan, ROC

Keywords: Timetable Scheduling, Genetic Algorithm, Tabu Search Approach

Abstract

In a timetabling problem, exam subjects must be slotted to certain times that satisfy several of constraints. They are NP-completeness problems, which usually lead to satisfactory but sub-optimal solutions. This paper investigates and compares Genetic Algorithm and Tabu Search approaches to solve these kinds of problem. The experiment shows that TS approach can produce better timetables than those of GA approach can. Further, the search time spent in TS is less than that of GA. However, GA can produce several different near optimal solutions simultaneously.

1. Introduction

Timetable Scheduling Problems are particularly challenging for Artificial Intelligence and Operations Research techniques. They are problems of time-based planning and combinatorial optimization that tend to be solved with a cooperation of search and heuristics, which usually lead to satisfactory but sub-optimal solutions. There are many kinds of timetable scheduling problems in the daily life, such as examination, lecture, transportation timetables. In general, there are several common difficulties in these kinds of problems:

- There exists no deterministic polynomial time algorithm; they are computationally NP-complete problems.
- It is very difficult to design effective heuristics; advanced search techniques using various heuristics to prune the search space will not be guaranteed to find an optimal solution.
- A general algorithmic approach to a problem may turn out to be inapplicable; certain special constraints are often required in a particular instance of that problem.
- Real world timetable scheduling problems often involve constraints that cannot be precisely represented or even stated.

In a timetable-scheduling problem, events, for example exam subjects, must be slotted to certain times which satisfy various of constraints. Knowledge-based and OR-based approaches to solving such problems are hard to develop, are often slow and can be inflexible because the architecture itself was based on specific assumptions about the nature of the problem. Constraint satisfaction techniques have been used to solve hard constraints; however, it is more difficult to handle soft constraints such as preferences. Most often people still resort to handcrafted solutions starting from an easier solution and making a sequence of modifications to cater for changed requirements. This typically leads to sub-optimal solutions that bring significant organizational or financial penalties. [1] developed a Genetic Algorithm (GA) based timetable system which has been successfully used by a university. This paper investigates the GA and Tabu Search (TS) approaches to schedule exam timetables, and further compares the performances of these two techniques. The main issues we are going to address are as follows:

- The quality of the examination timetable.
- The time spent in producing the timetable.

2. Problem Description

In this specific timetable-scheduling problem, the events are exams, and the times are separate time slots. The number of exams and students is 10 and 50 respectively. There are three days for the allocation

0-7803-5578-4/99/$10.00©1999 IEEE

of the exams. Each day contains four time slots, two in the morning and two in the afternoon. Table 1 illustrates the exam timetable framework used for this paper, where 1, 2, 3, ... 12 are time slots. A student may take different number of exams because they may take different number of courses. Here, we consider this problem in terms of number of exams at most per student in the following three cases:

- a student takes 7 exams at most,
- a student takes 8 exams at most, and
- a student takes 9 exams at most.

The most important constraint of the exam timetable is that the same student cannot take two different exams at the same time. In other words, the exams cannot clash for any student. This is the only hard constraint. However, we also consider the several soft constraints:

- Very strongly prefer no more than 2 exams per day for any student.
- Strongly prefer not to have exams consecutive on the same half day for a student.
- Prefer not to have exams consecutive on different half days of the same day for the same student.

Table 1. Time slots allocation for examination

Timetable	AM		PM	
	8:30~ 10:00	10:30~ 12:00	13:30~ 15:00	15:30~ 17:00
1st day	1	2	3	4
2nd day	5	6	7	8
3rd day	9	10	11	12

3. Genetic Algorithms Approach

GAs [2] are a group of methods that solve problems using the algorithms inspired by the processes of neo-Darwinian evolutionary theory. In a GA, each candidate solution to a problem (known as "chromosome") is typically an 0ordered fixed-length array of values (call "alleles") for attributes ("genes"). Each gene is regarded as atomic in what follows; the set of alleles for that gene is the set of values that the gene can possible take. The performance of a set of candidate solutions is first evaluated and ordered, then new candidate solutions are produced by selecting candidates as "parents" and applying *mutation* (slight alteration of the features of an existing "parent") or *crossover* (recombination of the features of a pair of "parent" solutions) operators. Both these operators essentially combine bits of two candidates to produce one or more "children". The children are thus a new set of candidates to be evaluated, and this evolutionary cycle is repeated from generation to generation until an adequate solution is found. An outline of the GAs is shown in Figure 1.

The representation that we used here is simply a list of numbers length e (the number of exams to be scheduled), each element of which is a number between 1 and t (the number of timeslots available). The interpretation of such a chromosome is that if the nth number in the list is t, then exam n is scheduled to occur at time t. For example, the chromosome, [8,11,6,1,2,5,1,2,3,6], is a candidate solution in which exam 1 takes place at time 8, exam 2 takes place at time 11, etc. [3,7,9,2,1,10,4,5,8,11] is another chromosome. We then randomly choose a point, say 3, and crossover them to produce two children as shown in Figure 2. The evolutionary cycle will repeat over and over again until an optimal timetable is found or a certain maximum number of generation is reached.

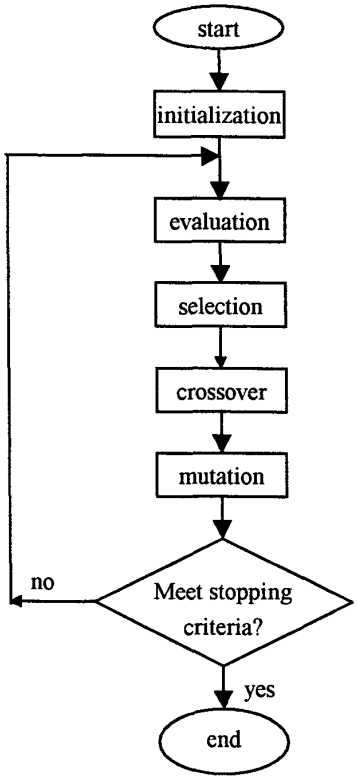
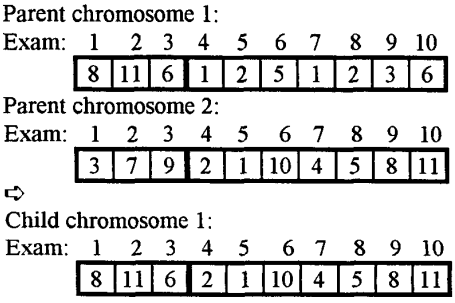


Figure 1. Outline of GAs



Child chromosome 2:

Exam:	1	2	3	4	5	6	7	8	9	10
	3	7	9	1	2	5	1	2	3	6

Figure 2. Representation and crossover

4. Tabu Search (TS) Approach

TS [3,4] is a kind of heuristic search method which has the advantage of having internal memory. This approach contains two basic mechanisms: *tabu restrictions* and *aspiration criteria*. Mapping to our paper, the former represents the best solutions found now have already been recorded in the *tabu list memory* before. The latter is similar to the former one, but its solution quality is the best in all the iterations found so far. So, they can be put into the *tabu list memory* again to test *tabu restrictions* in the next iteration.

Here, we used the same representation as our GA approach to encode the timetable. We produce a group of 20 candidate solutions from the best solution each iteration. We then pick the best candidate solution and mutate it by randomly changing two exams time slots (i.e., two memory values) at a time for ten times to produce the solutions for the next iteration. The best of the new solutions is then selected again to test by *tabu restrictions* and *aspiration criteria*. They may be put into the *tabu list memory* depending on whether they pass the former test. If they fail, but satisfy the latter test, they can also be put in the *tabu list memory*.

The advantage of TS over other methods is to use its memorized ability to prevent from searching previous visited areas. Therefore, it is easier to escape from local optimum and approach the global or near global optimum in a short time. Figure 3 is an outline of TS. Here f_x is the evaluation function for a timetable. Also, the timetables are sorted according to their quality. In this way, we can always pick the best one first to test from the beginning of the list.

5. Experiments and Results

The overall quality of the timetable is evaluated by the evaluation function that adds up all violations of all constraints by testing it with each student's exams. Each constraint has an associated 'weight' or 'penalty' defined in an intuitive way as follows. (1) A single clash has a penalty of 300. (2) An instance of four and three exams in one day has a penalty 4 and 3 respectively. (3) An instance of two exams consecutive in the same half day or in different half day has a penalty 2 and 1 respectively. The results are shown in Table 2, 3 and 4. In the Table 2 case, the average penalty of GA and TS is 84.6 and 40.4 respectively and the computing time is 2.082 and 1.870 seconds respectively.

In the Table 3 case, the former of GA and TS is 108.5 and 54.7 respectively and the latter is 1.934 and 1.857 seconds respectively. Finally, In the Table 4 case, the former is 146.3 and 99.6 respectively and the latter is 2.232 and 2.138 seconds respectively. These specific experiments show that TS can produces better solution, with less computing time, than those produced by GA. However, GA can produce several different near optimal solutions at the same time because of it holds the whole generation of chromosomes which may not originate from the same parents.

Table 2. Seven exams at most per student

seed	GA		TS	
	Penalty	Time(sec)	Penalty	Time(sec)
1	68	2.090	40	1.907
2	72	1.940	40	1.883
3	111	2.013	45	1.813
4	88	2.125	52	1.850
5	53	2.173	49	1.850
6	95	1.987	45	1.873
7	86	2.114	33	1.877
8	96	2.132	40	1.892
9	103	2.137	29	1.911
10	74	2.002	31	1.906
average	84.6	2.082	40.4	1.870

Table 3. Eight exams at most per student

seed	GA		TS	
	Penalty	Time(sec)	Penalty	Time(sec)
1	89	1.833	62	1.720
2	108	1.956	53	1.772
3	118	1.913	58	1.844
4	106	2.084	47	1.849
5	60	2.005	54	1.819
6	120	1.988	49	1.835
7	95	1.918	59	1.871
8	158	1.943	73	1.882
9	141	1.906	43	1.919
10	90	1.795	49	1.794
average	108.5	1.934	54.7	1.857

Table 4. Nine exams at most per student

seed	GA		TS	
	Penalty	Time(sec)	Penalty	Time(sec)
1	127	2.322	102	2.095
2	141	2.167	96	2.203
3	154	2.293	98	2.114
4	154	2.385	98	2.142
5	99	2.368	102	2.162
6	148	2.307	104	2.120
7	125	2.412	98	2.160
8	204	2.285	102	2.104
9	192	2.307	93	2.068
10	119	2.478	103	2.213
average	146.3	2.232	99.6	2.138

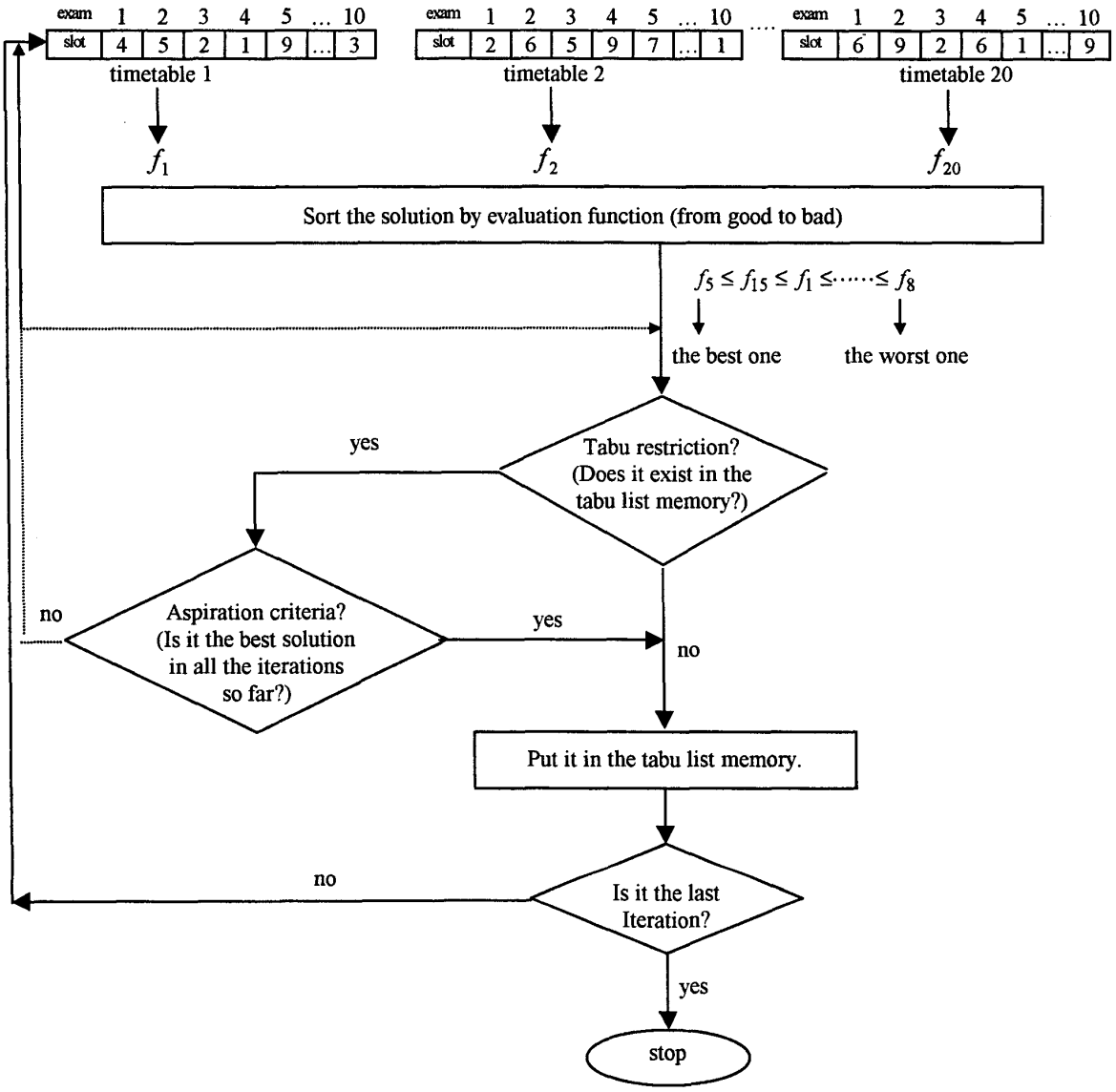


Figure 3. Outline of TS

References

[1] Fang, H. L., "Investigating Genetic Algorithms in Scheduling", Technical Paper 12, Department of Artificial Intelligence, University of Edinburgh, 1992.

[2] Holland, J. H., "Adaptation in Natural and Artificial Systems", University of Michigan Press, 1975.

[3] Glover, F., "Tabu Search, Part I " ORSA J. Computing, vol. 1, no. 3, 1989, pp. 190-206.

[4] Chu, S. C. and Pan, J. S., "Tabu Search Algorithms to VQ Codevector Index Assignment for Noisy Channels", Proceedings of The Sixth Australian International Conference on Speech Science and Technology (SST-96), 1996, pp. 169-174.