Solving the School Time Tabling Problem using Tabu Search, Simulated Annealing, Genetic and Branch & Bound Algorithms

Peter Wilke · Johannes Ostler

the date of receipt and acceptance should be inserted later

Abstract Using a real world School Time Tabling problem we compare the performance of different optimization algorithms, namely Tabu Search, Simulated Annealing, Genetic Algorithm and Branch & Bound. All experiments are being executed using the same problem specification. The used software was developed by the authors to compare several optimization algorithms under identical conditions.

Keywords School Time Tabling Problem, Tabu Search, Simulated Annealing, Genetic Algorithms, Branch & Bound Algorithms

1 Introduction

The global goal of our software development is to provide a software framework which is capable of solving various timetabling problems. When a problem is specified it is of particular interest what algorithm delivers the most suitable solution, indeed depending on the context.

We will show how computation time and solution quality differ for the school problem depending on the chosen algorithm.

Here we will focus on this problem. The authors are more than happy to answer any questions regarding the software used or to arrange for a test drive.

2 The School Time Tabling Problem

The School Time Tabling problem is well known, therefore we here list just the specifics of our case. The school is a German high school with 113 teachers, 100 rooms and 43 classes. The framework supports basically two kinds of constraints, namely hard

Peter Wilke
University Erlangen-Nuremberg, Informatik 2
Martensstrasse 3, 91058 Erlangen, Germany
Tel.: +49.9131.8527624
Fax: +49.9131.8528809
E-mail: wilke@cs.fau.de

and soft constraints. Some algorithms, like Branch & Bound, forbids the violation of a hard constraint. Algorithms, e.g. which uses neighborhood search techniques, allow the violation of hard constraints, because the path to a better solution can over a invalid solution lead. Therefore hard constraints must get higher weights then soft constraints.

The hard constraints of our school problem definition are time clashes for teachers, classes and rooms and the room size constraint. Time clashes are assigned 200 penalty points, while room size constraint violations cost 50 points. For every event and resource type is specified the minimal and maximal count of resources which should be assigned to this event. Dropping the minimum count or exceeding the maximum causes in each case penalty of 10 * diff points. Whereby diff = min - x respectively diff = x -max and x is the count of assigned resources. The search space is in several tests limited to solutions which comply with minimum and maximum constraint. That is the case by Simulated Annealing, Tabu Search and Genetic Algorithm.

3 Test Settings

Every algorithm has been run 10 times on a Laptop with a Intel Core Duo 1.6 GHz and 2.0 GB RAM, running Linux with kernel 2.6.17.

3.1 Tabu Search

Tabu Search (TS) finds improvements for about 6 minutes, which is equivalent to approx. 6,000 iterations. Tests have shown that not significant improvement was found after 10 minutes or more. Because TS is able to find better solution until the end of the computation, we decided to perform 200 moves within 1 iteration, e.g. keep the number of iterations quite high and the number of moves in an iteration step low. The best regular tabu list length seems to be approx. 40 elements. For diversification the tabu list length was extended to 300 for 4,000 iterations when no improvement was found after 10,000 iterations.

3.2 Simulated Annealing

In our experiments we decided to use a reduction factor of 0.9 and an initial acceptance probability of 0.8 to cool down quite slow. As the three hard constraints of the School Problem differ quite a lot in there impact we used 3,000 moves to determine the costs of two neighbour. A steady state is reached by a quite high level of coolingConst = 1,400. In case a move improves the solution cooling is 10 times (factorReconf=10, factorSucessfulReconf=1) faster as otherwise. The cooling factor decreaseFactor is set to 0.9. The computation ends after maximal 20 Million iterations respectively 2 hours or if a plan with zero costs is found.

3.3 Genetic Algorithm

Our Genetic Algorithm uses 30 individuals and runs at most 2 hours i.e. 7,200 seconds. The best individual of a generation will survive and 5 % of the individuals. The other will be generated by crossover over two individuals. Every resource list of the individual is subject to mutation with a probability of 0.5 %.

3.4 Branch & Bound

We were interested in how an algorithm able to find the global optimum would perform compared to the local optimum search algorithms described above. So we implemented Branch & Bound as a reference algorithm. If there would be endless computation time spent, Branch & Bound would find the best result. But in real world time is limited. So we spend only 8 hours for the computation. Because of the peculiarities of the algorithm minimum constraints must not be a hard constraint. So minimum constraints could be violated.

4 Summary

Tabu Search requires compared to the other algorithm only a short time to find good solutions. It finds them quite fast and improves until the very end. It is remarkable the Tabu Search in none of the runs even violated a Clash Constraint. The Room Size Constraint was violated once every time, so no correct solutions were found by Tabu Search. The initial solution was improved by 99.9 % in a 6 minutes run. The best solution was found after 100 seconds.

Tabu Search find good solution quicker than Simulated Annealing, e.g. Tabu finds a solution with 50 penalty points in 347 sec, while Simulated Annealing has found a solution with 1050 penalty points at this point in time, but is able to reduce costs to an optimum of 0 further on. Simulated Annealing is slower than Tabu Search, it's solution become better after approx. 450 seconds. Simulated Annealing beat Tabu Search in best and average performance. All generated time tables are valid and do not violate any hard constraints. The initial solution is improved by perfect 100% in 8 minutes run time.

The Genetic Algorithm was able to improve the best found solution during the whole 2 hours run time. There might be a chance that this would continue if more compute time would be spent. In our experiments all generated time tables violate one or more hard constraints, e.g. no valid time table was produced. Compared to Tabu Search the violation caused by Genetic Algorithm are even more severe. In 2 hours the Genetic Algorithm was able to improve the initial solution by 99%.

Branch & Bound required the longest computation time - as expected. After 8 hours the best solution 3300 penalty points, which is quite expensive, the improvement was 79%, which is in some degree good. Branch & Bound violates only but always the Minimum Constraint, i.e. no valid solutions were generated. In best solution Minimum Constraint was violated 128 times.

Figure 1 shows the time - cost chart of the computation of the best solutions for each algorithm.

The solution of Simulated Annealing violates no constraint, Branch & Bound 128 times the Min Constraint, figure 2 compares the solution quality of Tabu Search and Genetic Algorithm.

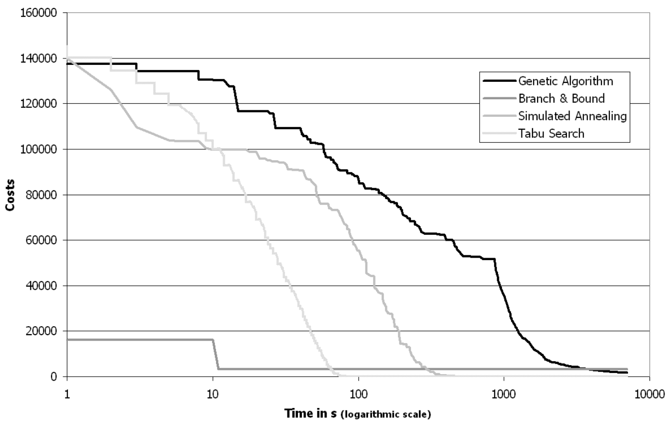

Fig. 1 Time - cost chart of Genetic Algorithm, Branch & Bound, Tabu Search and Simulated Annealing in comparison

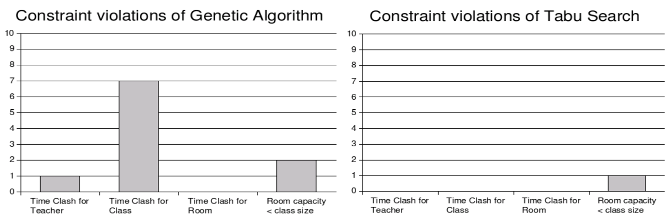

Fig. 2 Constraint violations caused by Tabu Search and Genetic Algorithm

In the given circumstances we would recommend Simulated Annealing to generate the School Time tables, because it is the best tradeoff between execution time and quality of result.

Acknowledgements We would like to thank all members of our time tabling software development group for their contributions. Special thanks to Sabine Helwig and Sabeur Zarai for implementing the school problem in various versions. All trademarks are property of their registered owners.

References

1. Stefan Büttcher. Entwurf und Implementierung eines Genetischen Algorithmus zur Erstellung von Zeitplänen. Studienarbeit, Universität Erlangen–Nürnberg, 2002.
2. M. Gröbner and P. Wilke. A general view on timetabling problems. In Edmund Burke and Patrick De Causmaecker, editors, *Proc. of the 4th International Conference on the Practice and Theory of Automated Timetabling*, pages pp. 221–227. PATAT 2002, August 2002.
3. Matthias Gröbner. Ein Modell zur Beschreibung und Lösung von Zeitplanungsproblemen. Dissertation, Universität Erlangen–Nürnberg, 2003.
4. P. Wilke, M. Gröbner, and N. Oster. A hybrid genetic algorithm for school timetabling. In *Proc. AI2002 15th Australian Joint Conference on Artificial Intelligence*, 2002.