A Hybrid Simulated Annealing-Tabu Search Algorithm for Post Enrolment Course Timetabling

Juan Frausto-Solís, Federico Alonso-Pecina

Tecnológico de Monterrey, Campus Cuernavaca,
62790 Xochitepec, Morelos, México
juan.frausto@itesm.mx, federicoalonsopecina@hotmail.com

Abstract. Course Timetabling is a well known NP hard problem. This paper presents a hybrid Simulated Annealing-Tabu Search algorithm named Fralonso, which was developed for the Course timetabling problem of Patat track2. This algorithm is designed to find feasible solutions. Fralonso has two phases; the first one finds a feasible solution which is improved in the second one. In every phase Tabu Search is used after Simulated Annealing algorithm in order to refine the search process. In addition, Fralonso uses two neighborhood schemes to preserve a good diversity level.

1 Introduction

Fralonso is a hybrid algorithm of Simulated Annealing (SA) and Tabu Search (TS) developed for the International Timetabling competition 2007 [1]. The track of this algorithm was the Post Enrolment Course timetabling problem (track2). The Problem Description and instances can be found in [1], and for the sake of brevity we do not repeat them here. Fralonso has two phases: the first one looks for a feasible solution by using three procedures, two of them based on Simulated Annealing (SA) algorithm and the other based on Tabu Search (TS). If a feasible solution is found during the first phase, the second phase is executed to improve the feasible solution of the first phase. This paper describes Fralonso and the results obtained.

3 Algorithm Description

Fralonso is shown in figure 1.The first phase starts with a SA named SAET (Simulated Annealing Extra Timeslots) which adds extra timeslots hoping to find a FS more easily. Actually, this strategy has been successfully used for hard instances in other researches as [4]; particularly it was used in some algorithms participating in the first PATAT competition [5]. IF a FS is not found, then other SA (named SAF, Simulated Annealing Feasible) is executed in order to search a FS without extra timeslots. Otherwise, if a FS is found the second phase continues with a second SA named SAO (Simulated Annealing Optimum), which searches to reach a solution as close to the optimum as possible. Probably SAO finishes with the optimal solution or more probably not; in the last case a TS procedure named TSO searches for the optimal one. Again TSO probably finishes with the optimal or more probably not. When SAET finds a FS, Fralonso executes the second phase trying to improve it; this is done by reducing the number of violated soft constraints. In the second phase, a SA-TS algorithm is executed as follows: first a SA is executed, searching in the solution space but without extra timeslots. The stop criterion of this SA is its frozen temperature or the limit of PATAT´s competition. If this frozen temperature is reached before the limit of time then a TS is executed as a local search.

3.1 Initialization

The solution is structured as a matrix which dimensions are the number of rooms per number of periods. To initialize the solution every event is placed without violating any hard constraint in only a unique room-timeslot combination. In order to achieve this goal some events can be placed in extra timeslots. The two last hard constraints are ignored for the events which are placed in the extra timeslots. After this initialization SAET is executed.

3.2 Simulated Annealing Extra Timeslots (SAET)

In SAET, the objective function (energy of SA algorithms) is the number of students programmed in extra timeslots. That means we are minimizing the number of violations on extra timeslots; for instance, if a student has three events placed in extra timeslots, then three violations are counted. At the beginning of any SA, the temperature parameter is measured in the same units as the energy in SA; therefore, at the beginning of the first SA the initial temperature is tuned as the number of events minus 45 (because 45 is the number of normal periods, i.e. without time slots). The SA algorithm in SAET is tuned using a Markov approach previously published in [3], but the maximum length of the Markov chain is set differently, and it was tuned experimentally (its value is 10000). The neighborhood of SA is defined as follows: for a given solution a neighbor event is chosen at random making a simple movement which does not violate any hard constraint. In other words, a feasible neighborhood is determined by making movements: i) which do not produce any violation to the hard constraints for the normal periods or ii) they do not introduce any violation in the first three hard constraints for all the extra periods. SAET is stopped until the temperature cycle is frozen (the temperature is equal to 0.01) or a FS is found. If a FS is found by SAET SA is input for the second phase.

3.2 Simulated Annealing to find a Feasible solution (SAF)

SAF is called when SAET does not find a FS; in this case, extra periods are compressed by a function called Compressed_Solution (CS). The function CS only reallocates the events that are located into extras periods to the spaces of the first 45 periods, even though this violates hard constraints. The initial temperature of SAF is set to 470 * number of students + number of events. The cooling scheme of SAF follows a geometrical function: T(k+1)= α*T(k). The alpha value is set as 0.99 and the SAF stop criterion is very simple: when the temperature reaches 0.01 or a feasible solution is found. In the metropolis cycle, a random event is chosen at the beginning, then a feasible neighborhood is calculated (only with movements that do not introduce a new violation on the hard constraints). If at the end of the metropolis cycle, the global solution is not improved, then the best SAF solution will be the input of a TS trying to find a feasible solution (TSF). The TSF tenure is a random number between one and twenty. The best TSF solution is returned to SAF and the process is repeated until a feasible solution is found or the limit of time is reached. If a feasible solution is found for any process of the first phase, this one will be the input of the second phase, which starts with a Simulated Annealing process trying to Optimize the solution (SAO). This phase will be described in the next section.

3.3 Simulated Annealing to Optimize the solution (SAO)

The initial temperature of SAO is set to 470 * number of students + number of events. The cooling scheme of SAO is geometric with an alpha value of 0.99 and the system is frozen with a very sample criterion: when the temperature reaches 0.01, or when ten metropolis cycles in a row give the same solution, or the time limit is reached. The cycles with the same solution in a row will be counted only when the temperature will be less or equal to –8 * Maximum capacity of rooms / log (0.95). The Markov chain of SAO is tuned using a scheme published in [3]. The maximum length of the Markov chain is set to 10000. The neighborhood of SAO is similar to SAET: for a given solution a neighbor event is chosen at random making a simple movement which does not violate any hard constraint. The output of SAO is the input of a TS which tries to optimize the latter solution (this TS is named TSO).

3.4 Tabu Search to Optimize the solution (TSO)

For this TS, two neighborhoods called N1 and N2 are implemented: I) N1 neighborhood: for a given solution, two steps to get a neighbor should be implemented. These steps are: a) Two events are interchanged and then b) A movement of an event to an available place is carried out. II) N2 neighborhood: A neighbor is found using the two steps of N1 neighborhood and then applying the next step: c) Interchange all the events of two periods k_1 and k_2 where $k_1 \neq k_2$. This implementation of TS seeks to improve the objective function. Neither of these neighborhoods violate any hard constraint. Where a movement improves the best found solution then the movement is done, and the tabu status is ignored, where a movement without the tabu status improves the actual solution then the movement is also done. Otherwise, a random movement of the event with less frequency changes and without status tabu is done. In all the movements executed, the events are put in the tabu list. Every time an event changes its room or period, its frequency is incremented by one. The tenure in the tabu list is a random number b $\in \mathbb{Z}^+$ ($20 \leq b < 40$). In the TS proposed here, the stop criterion is as follows: a) The optimum solution (zero hard constraints violated and zero soft constraints violated) is found or b) Maximum number of iterations without improving the last best solution is reached (100 iterations were used in this work), or c) the limit of time is reached. Finally, if any FS is found, then all the events implied in any hard constraint are set in a room and period labeled as "-1".

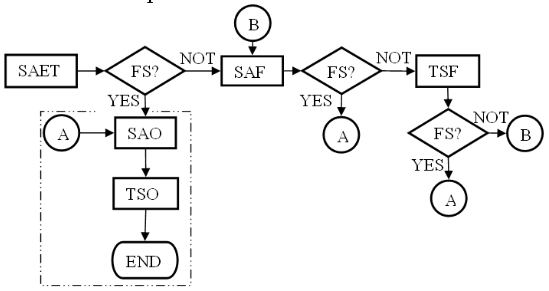

Fig. 1. Diagram of flow of FRALON

4 Results and Conclusions

The algorithm was run fifty times per instance; in average the algorithm reaches 89.375% of FSs. In every instance the algorithm was able to find a FS. In the first stage of the contest, Fralonso obtained the seventh place. In the results of PATAT for Post Enrolment Course Timetabling, where only the first sixteen instances are used, the order remained unchangeable. Ten random runs per instance of Fralonso were selected to compare with the five finalists. In this case, the algorithm would obtain the sixth place. However, it would be in second place for finding feasible solutions, and it would only be behind the algorithm of Chiarandini et al. The future work will be to improve Fralonso, mainly in the optimization part.

References

1. International Timetabling Competition 2007 http://www.cs.qub.ac.uk/itc2007/ date of consultant February of 2008.
2. International Timetabling Competition, URL: http://www.idsia.ch/Files/ttcomp2002/ Consultant date: Wednesday , November 28 of 2007
3. Sanvicente-Sanchez, H. and Frausto, J. (2004). Method to Establish the Cooling Scheme in Simulated Annealing Like Algorithms. International Conference, Assis, Italy. ICCSA (2004) LNCS Vol. 3095. 755-763.
4. Lewis R., Paechter B., Finding Feasible Timetables using Group-Based Operators. Technical Report NUS-2005-01, IEEE Transactions on Evolutionary Computation.
5. Socha Krzysztof. "MAX-MIN Ant Systems for International Timetabling Competition", March 31, 2003. International Timetabling Competition.