An Evolutionary Algorithm for High School Timetabling

Jonathan Domrös · Jörg Homberger

Abstract An Evolutionary Algorithm for high school timetabling problems, which is used for the Third International Timetabling Competition (ITC2011), is presented. The solver is based on two ideas: Firstly, an indirect representation of timetables is used. Each coded solution consists of a permutation of sub-events. Secondly, the evolutionary search is controlled by the population concept of the (1+1)-Evolution Strategy.

Keywords High school timetabling · International timetabling competition · Evolutionary algorithm

1 Introduction

Evolutionary Algorithms are metaheuristics which are often based on a coding of decision variables or solutions (e.g., Miettinen et al., 1999). They have been used successfully for (high) school timetabling so far (e.g., Burke and Newall, 1999; Bufe et al., 2001; Beligiannis et al., 2009; Raghavjee and Pillay, 2010). However, most approaches have been developed and evaluated for high school timetabling problems with a smaller number of constraints than the problems used for the ITC2011.

The high school timetabling problems of the ITC2011 are described in Post et al. (2012). In order to describe our algorithm, especially our coding, it is important to know, that the problems take up to three different kinds of problem decisions into account: (1) split-decisions (for a given event the number of sub-events and the duration of each sub-event have to be calculated); (2) time-decisions (for a sub-event a starting time has to be assigned); and (3) resource-decisions (for a sub-event one or more event resources have to be assigned).

2 Overview of the Evolutionary Algorithm

An overview of the developed Evolutionary Algorithm is shown in Fig. 1.

J. Homberger
University of Applied Sciences Stuttgart, Schellingstr. 24, 70174 Stuttgart, Germany
Tel.: +49 711 8926 2511
Fax.: +49 711 8926 2553
E-mail: joerg.homberger@hft-stuttgart.de

(1) **preprocessing:** calculate all feasible sub-events;
(2) **initialization**: calculate one coded timetable ***C*** randomly on the basis of (1);
(3) **decoding**: construct a timetable ***T*** by decoding ***C***;
(4) **evaluation:** calculate the infeasibility value and the objective value for ***T***;
WHILE (NOT a given computational time limit has been reached)
(5) **mutation:** calculate a new coded timetable ***C**** by mutation of ***C***;
(6) **decoding**: construct a new timetable ***T**** by decoding ***C****;
(7) **evaluation:** calculate the infeasibility value and the objective value for ***T****;
(8) **replacement**: IF(***T**** is better than ***T***) THEN (***C*** := ***C****; ***T*** := ***T****);
(9) **output**: the timetable ***T***;

Fig. 1 The Evolutionary Algorithm for High School Timetabling

As shown in Fig. 1, the population concept of the (1+1)-Evolution Strategy, which is simply a randomized hill-climbing method, is used in order to control the search (e.g., Beyer and Schwefel, 2002; Mester and Bräysy, 2004). That means, in each iteration one solution or timetable T^* is calculated by mutation of the current best solution T. The latter is replaced by T^*, if T^* is better than the current best solution T (for the evaluation of timetables see Post et al., 2012). The mutation does not directly work on timetables. Instead, the mutation varies a coded version C of the current best timetable T. The coding of solutions and the steps of the algorithm are explained in the next sections.

3 Coding of solutions and preprocessing

Solutions are coded by a permutation of eligible sub-events or, more precisely, a permutation of the indices of eligible sub-events. The aim of preprocessing is to calculate the set of eligible sub-events, taking split events constraints, distribute split events constraints, and link events constraints into account. Therefore, for each given event i and its given duration D_i all feasible splits are calculated. A split of an event i is a division into one or more sub-events with individual durations. A split is feasible, if the split events constraints and the distribute split events constraints are fulfilled for i. The number of feasible splits of event i is denoted by n_i. Each sub-event $s_j = (i_j, k_j, d_j)$ can be described by a triple, which consists of the corresponding event i_j, the corresponding split k_j ($k_j = 1, ..., n_i$), and a sub-event duration d_j. The total duration of all sub-events, which corresponds to the same split of an event i is equal to D_i. Fig. 2 describes the prepossessing procedure by an example. The example takes the split events constraints into account, i.e., a minimum and a maximum duration for sub-events, and a minimum and a maximum amount of sub-events have to be considered, when eligible sub-events are calculated.

Link events constraints could reduce the set of feasible splits. In case that two events i and l must be executed in parallel, the corresponding splits must be "compatible", i.e., for each sub-event s_m of event i exists one sub-event s_n of event l, such that $d_m = d_n$. Thus, each feasible split of i (l) with no compatible split of l (i) is deleted from the set of feasible splits.

As mentioned, each coded solution is just a permutation of the indices of the calculated eligible sub-events. In our example of Fig. 2 each coded solution is a permutation of the indices 0, ..., 7.

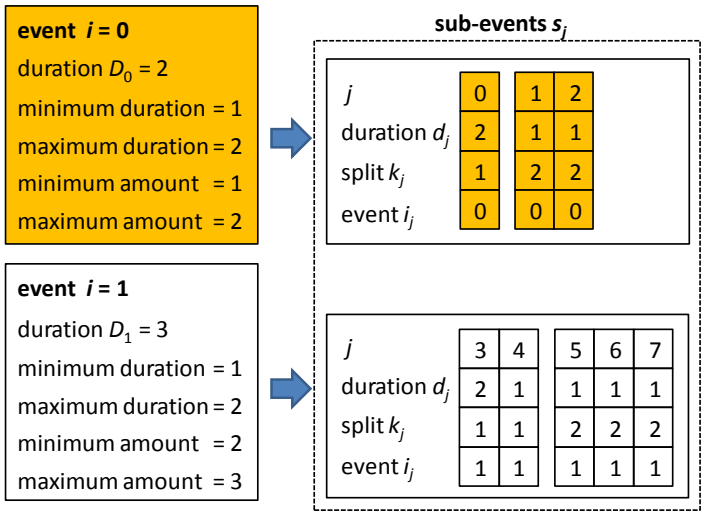

Fig. 2 Preprocessing procedure

4 Initialization and mutation

At the beginning of the search (step 2) one coded solution C^* is constructed randomly, i.e., a random permutation of the indices of the eligible sub-events is calculated. During mutation (step 5) this permutation is varied randomly, i.e., two indices are randomly chosen and then swapped. The swap-mutation is often suggested for permutation based Evolutionary Algorithms (e.g., Gottlieb, 2000).

5 Decoding

The decoding procedure aims to construct a feasible timetable T on the basis of a coded solution C. A timetable T consists of sub-events, which are extended by starting times and event resources. The decoding method is very complex, since it considers all constraints of the problem at hand in order to find a feasible solution. In the following, we just present some of the basic ideas of the procedure.

To construct a timetable T, three steps are executed within each iteration of the iterative decoding procedure: (*i*) One eligible sub-event $s_j = (i_j, k_j, d_j)$ is selected; (*ii*) the split k_j is checked to be "practicable"; (*iii*) in the positive case (k_j is a practicable split), an insertion heuristic is executed in order to assign a starting time and all required event resources to s_j.

The order to select sub-events in step (*i*) is based on the order of indices in the coded solution C. If the index j of sub-event s_j is ordered more left than the index q of sub-event s_q, s_j is selected before s_q and thus has a higher priority to be inserted into the timetable T.

A split k_j is defined as "practicable" in step (*ii*), if no sub-event of the corresponding event i_j has been inserted into T or if only sub-events of split k_j have been inserted into T so far. The check of practicability is necessary, since for each event only sub-events of the same split can be inserted into a timetable.

The insertion heuristic of step (*iii*) checks, if at least one assignment of a starting time and of event resources for s_j exists, such that all hard constraints are satisfied. In a positive case, the heuristic inserts s_j into T, i.e., assigns a starting time and event resources such that no hard constraint is injured (time and resource decisions are made). If s_j is the first sub-event of the event i_j, which is inserted

into T, the corresponding split k_j is selected for i_j (split-decision is made). In the case, that no such feasible assignment of time and resources exists, the sub-event s_j is not inserted into T.

There is one exception from the rule that in each iteration of the decoding procedure only one sub-event (s_j) is selected and inserted into T. In case of link events constraints, where one or more events exist, which should be placed in parallel to i_j, sub-events of these events are also selected in step (*i*) and inserted together with s_j in step (*iii*).

It should be remarked, that sub-events of an event are not inserted at all, if the insertion would violate hard constraints. However, not inserting an event violates assign times constraints. The decoding method could be improved by executing an additional construction step (at the end of decoding) which inserts all these events, which were not inserted so far, in order to reduce the infeasibility value.

6 Conclusion

The developed Evolutionary Algorithm is suitable to solve the problems of the ITC2011. Our solver was a finalist in ITC2011. However, the solutions generated in round 1 are not a match to the existing best known or optimal solutions.

References

Beligiannis GN, Moschopoulos CN, Likothanassis SD (2009) A genetic algorithm approach to school timetabling. Journal of the Operational Research Society 60(1), 23–42

Beyer HG, Schwefel HP (2002) Evolution strategies: a comprehensive introduction. Journal Natural Computing 1(1), 3–52

Bufe M, Fischer T, Gubbels H, Hacker C, Hasprich O, Scheibel C, Weicker K, Weicker N, Wenig M, Wolfangel C (2001) Automated solution of a highly constrained school timetabling problem – preliminary results. In: Proc. of the Evoworkshops on Applications of Evolutionary Computing, Springer, pp 431–440

Burke EK, Newall JP (1999) A multi-stage evolutionary algorithm for the timetable problem. IEEE Transactions on Evolutionary Computation 3(1), 63–74

Gottlieb J (2000) Permutation-based evolutionary algorithms for multidimensional knapsack problems. In: Proceedings of the 2000 ACM Symposium on Applied Computing – Vol. 1, ACM New York, NY, USA, pp 408–414

Mester D, Bräysy O (2007) Active-guided evolution strategies for large-scale capacitated vehicle routing problems. Computers and Operations Research 34, 2964–2975

Miettinen K, Mäkelä MM, Neittaanmäki P, Périaux J (eds), Evolutionary algorithms in engineering and computer science. Chichester et al., 1999

Post G, Gaspero LD, Kingston JH, McCollum B, Schaerf A (2012) The third international timetabling competition. In: Proceedings of the Ninth International Conference on the Practice and Theory of Automated Timetabling (PATAT 2012), Son, Norway, August 2012

Raghavjee R, Pillay N (2010) An informed genetic algorithm for the high school timetabling problem. In: Proceedings of the 2010 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists, pp 408–412