Hybridizing Integer Programming and Metaheuristics for Solving High School Timetabling

Sørensen, Matias; Stidsen, Thomas Jacob Riis

Published in:
Proceedings of the 10th International Conference on the Practice and Theory of Automated Timetabling

Publication date:
2014

Citation (APA):
Sørensen, M., & Stidsen, T. R. (2014). Hybridizing Integer Programming and Metaheuristics for Solving High School Timetabling. In Proceedings of the 10th International Conference on the Practice and Theory of Automated Timetabling (pp. 557-560)

Hybridizing Integer Programming and Metaheuristics for Solving High School Timetabling

Matias Sørensen · Thomas R. Stidsen

Keywords High School Timetabling · Integer Programming · XHSTT · Metaheuristics · Matheuristics

1 Introduction

The preferred solution methodology within the area of high school timetabling is *heuristics* and the sub-field of *meta-heuristics*. *Integer Programming* (IP) is generally believed to be less efficient, and many researchers seem to apply heuristics per default when facing a *High School Timetabling Problem* (HSTP). The hybridization of IP and metaheuristics has been studied for many decades, but has recently attracted attention under the alias of *matheuristics* (by the contraction of Mathematical Optimization and Metaheuristics). For instance, Ryan (2012) argues that it is time to *enjoy the best from both worlds*. See also Maniezzo et al (2009a) and Maniezzo et al (2009b) for some recent collections of work within this area.

In this text it will be shown that also for the HSTP it can be beneficial to apply matheuristics. Specifically, the general format XHSTT is considered (see Post et al (2012) for a description of the format). This format can be used for both exchanging models and solutions for the HSTP, and currently there are around 50 instances publicly available (originating from 12 different countries). As a test-bed for computational results, instances from the *International Timetabling Competition 2011* (ITC2011) (Post et al 2013) are used.

2 Hybridizing Integer Programming and Metaheuristics

Caserta and Voss (2010) describe the hybridization of exact methods and metaheuristics in terms of a master-slave structure. That is, either (I) the

Matias Sørensen
MaCom A/S, DK
Tel.: +45 3379 7900
E-mail: ms@macom.dk

Thomas R. Stidsen
Management Science
Department of Management Engineering
Technical University Of Denmark

metaheuristic acts as master at a higher level and controls the exact approach or (II) the exact method acts as the master and controls the metaheuristic. In this text, algorithms of type (I) are considered, and the exact method consists of an IP model solved using a generic IP-solver. Thereby the exact method can be thought of as a *local search* component, and the metaheuristic guides the local search on an overall level.

The IP model for the XHSTT format, which is used as a basis for the matheuristic, was recently developed in-part by the present authors. This IP model is capable of handling any instance of the XHSTT format, and therefore the same applies for the presented matheuristic. The matheuristic is described in the following.

Given is a problem instance P and set of neighborhoods N. The set of decision-variables $\mathcal{X}$ (which are part of the IP model) describes a solution to the problem instance. The basic idea of the algorithm is to modify a subset of variables $V \subseteq \mathcal{X}$ in each iteration of the algorithm (which is done in practice by invoking the IP-solver). Each neighborhood $n \in N$ defines a certain way of selecting V (possibly in a problem-dependent way), incorporating some element of randomness. This element of randomness is incorporated to help the algorithm escape local optima. With each neighborhood $n \in N$ is associated a size-parameter, which determines the amount of variables to select. This size-parameter is adjusted throughout the algorithm, based on how hard the neighborhood is to handle for the IP-solver. The idea of this adaptive layer is to obtain neighborhoods which are efficiently handled by the IP-solver.

The pseudo-code for the matheuristic is shown in Algorithm 1. In Line 3, an initial solution is constructed using a simple heuristic (in this case a greedy algorithm). In Line 5, a neighborhood is chosen. This selection is biased towards neighborhoods which have previously performed well, according to some measure. In lines 6 and 7 the neighborhood is applied, by selecting the set of variables to fix, and thereby the set of variables which are left free for the IP-solver to modify. In Line 8 the IP-solver is invoked. In case the IP-solver is unable to improve the current solution, the current solution is maintained. In Line 9 the adaptive layer of the algorithm adjusts the size of the chosen neighborhood, according to the final IP-gap found by the IP-solver. Line 10 un-fixes variables (in preparation for the next iteration of the algorithm).

Algorithm 1 Matheuristic pseudo-code

```
 1: input: XHSTT problem instance $P$, neighborhoods $N$
 2: output: feasible solution $S$
 3: $S :=$ construct initial solution of $P$
 4: while stopping criteria not met do
 5:     choose neighborhood $n$
 6:     obtain variables $V := n(S)$
 7:     fix variables $\mathcal{X} \setminus V$ to their current value
 8:     invoke IP-solver on $S$ with short timelimit
 9:     adjust size of $n$ subject to the obtained IP-gap
10:     unfix all variables
11: end while
```

Future research will investigate more advanced approaches for adjusting the size of the neighborhoods, and more advanced ways of selecting variables in each neighborhood. Such approaches will hopefully lead to more efficient algorithms.

3 Preliminary Results

For establishing computational results, Gurobi 5.5 is used as IP-solver. This is a commercial product, which is among the best IP solvers currently available. The XHSTT instances of round 2 of ITC2011 are used as test-bed. In this round of the competition, the performance of the finalist algorithms were compared on the same computer using the same time-limit (1000s). The ITC2011 organizers have released a benchmark executable which can be used for determining the equivalent runtime on any given computer. This gives us the opportunity to perform a fair comparison with the ITC2011 round 2 finalists, apart from the fact that the use of commercial software was not allowed in ITC2011. Nevertheless, the presented test-setup provides a good setting for showing the potential of the matheuristic. Table 1 shows the obtained results. These preliminary results show that the matheuristic finds the best solution on 9 of the 18 instances. Furthermore the matheuristic achieves second place when considering average ranks of the algorithms. These are promising results, and show the potential of hybridizing mathematical programming and metaheuristics.

Table 1 Performance of the matheuristic compared to the finalists of round 2 of ITC2011 and the IP solved directly with an IP-solver. For each instance is listed the average solution found from each of the competitors of round 2 of ITC2011 (columns *GOAL*, *HySTT*, *Lectio* and *HFT*), the performance of the IP when solved directly using Gurobi (column *IP*) and the performance of the matheuristic (the average solution obtained over 10 runs). An objective of a solution to a XHSTT instance is denoted (H, S), where H and S denote the cost of violation of the hard-constraints and the soft-constraints, respectively. The best solutions are marked in **bold**. Row *Avg. Ranks* denotes the average ranking of each solution method, 1 being best.

		GOAL	HySST	Lectio	HFT	IP	Matheuristic
BR	Instance2	(1, 62)	(1, 77)	38	(6, 190)	46	**6**
BR	Instance3	124	118	152	(30, 283)	39	**27**
BR	Instance4	(17, 98)	(4, 231)	(2, 199)	(67, 237)	(5, 286)	**58**
BR	Instance6	(4, 227)	(3, 269)	230	(23, 390)	682	**57**
FI	ElementarySchool	4	(1, 4)	**3**	(30, 73)	**3**	**3**
FI	SecondarySchool2	**1**	23	34	(31, 1628)	(1604, 3878)	6
GR	Aigio	**13**	(2, 470)	1062	(50, 3165)	(1074, 3573)	180
IT	Instance4	454	6926	651	(263, 6379)	17842	**48**
XK	Instance1	(59, 9864)	(1103, 14890)	(275, 7141)	(989, 39670)	(3626, 2620)	**(9, 23525)**
NL	Kottenpark2003	**90928**	(1, 56462)	(50, 69773)	(209, 84115)	(8491, 6920)	(238, 43143)
NL	Kottenpark2005A	**(31, 32108)**	(32, 30445)	(350, 91566)	(403, 46373)	(2567, 53)	(566, 19968)
NL	Kottenpark2008	**(13, 33111)**	(141, 89350)	(209, 98663)	-	(14727, 5492)	(6112, 353671)
NL	Kottenpark2009	**(28, 12032)**	(38, 93269)	(128, 93634)	(345, 99999)	(17512, 140)	(9418, 705605)
ZA	Woodlands2009	(2, 14)	(2, 70)	**(1, 107)**	(62, 338)	(1801, 705)	(2, 429)
ES	SpainSchool	894	1668	2720	(65, 13653)	(1454, 11020)	**485**
GR	WesternGreece3	**6**	11	(30, 2)	(15, 190)	25	**6**
GR	WesternGreece4	**7**	21	(36, 95)	(237, 281)	81	12
GR	WesternGreece5	**0**	4	(4, 19)	(11, 158)	15	**0**
Avg.	Ranks	2.2	3.4	3.3	5.3	4.6	2.3

References

Caserta M, Voss S (2010) Metaheuristics: Intelligent problem solving. In: Maniezzo V, Stützle T, Voss S (eds) Matheuristics, Annals of Information Systems, vol 10, Springer US, pp 1–38

Maniezzo V, Stützle T, Voß S (2009a) Matheuristics: Hybridizing metaheuristics and mathematical programming. Annals of Information Systems 10

Maniezzo V, Voss S, Hansen P (2009b) Special issue on mathematical contributions to metaheuristics. Journal of Heuristics 15(3)

Post G, Ahmadi S, Daskalaki S, Kingston J, Kyngas J, Nurmi C, Ranson D (2012) An xml format for benchmarks in high school timetabling. Annals of Operations Research 194:385–397

Post G, Gaspero L, Kingston J, McCollum B, Schaerf A (2013) The third international timetabling competition. Annals of Operations Research

Ryan D (2012) It is time to enjoy the best of both worlds. In: The 46th ORSNZ Conference, Victoria University of Wellington, New Zealand