INSTITUT FÜR INFORMATIK
Lehr- und Forschungseinheit für
Programmier- und Modellierungssprachen
Oettingenstraße 67, D–80538 München

Ludwig-Maximilians-Universität München

Constraint-Based Heuristics for Grammar School Timetabling

Slim Abdennadher, Michael Marte

appeared in *Workshop on Integration of AI and OR techniques in Constraint Programming for Combinatorial Optimization Problems, CP-AI-OR'99, Ferrara, Italy, 1999*
http://www.pms.informatik.uni-muenchen.de/publikationen
Forschungsbericht/Research Report PMS-FB-1999-1, January 1999

Constraint-Based Heuristics for Grammar School Timetabling

Slim Abdennadher and Michael Marte
Computer Science Department, University of Munich
Oettingenstr. 67, 80538 Munich, Germany
phone: +49 89 2178 2216
fax: +49 89 2178 2211
{Slim.Abdennadher, Michael.Marte}@informatik.uni-muenchen.de

1 Introduction

Constraint Logic Programming (CLP) [JM94, Wal96, FA97, MS98] is a rapidly growing research area aiming at the solution of large combinatorial problems. CLP combines the advantages of logic programming and constraint solving techniques. The use of CLP has the advantage that the solving procedure can easily be adapted to changing scheduling characteristics. The grammar school timetabling problem can be elegantly formalized as a constraint satisfaction problem and implemented by means of specialized constraint solving techniques that are available in CLP languages. The task of grammar school timetabling consists in scheduling classes, teachers and rooms into a fixed number of periods, in such a way that no teacher, class or room is used more than once per period. A lot of requirements are to be met which makes timetabling a difficult and time-consuming expert task.

In this paper, the generation of timetables for a German grammar school is tackled using the CLP framework. Our prototype has been implemented in collaboration with the Maria-Theresia-Gymnasium at Augsburg using Sicstus Prolog [Int98]. Because of the incomplete constraint propagation methods used for scheduling problems, the application programmer often has to explicitly use a labeling phase in which a backtracking search blindly tries different values for the variables. Since uninformed labeling may be expensive, the programmer needs to employ techniques for reducing the search space. There is a variety of techniques to do this. For our application, we integrated several heuristics for variable and value selection into the constraint solving environment of Sicstus Prolog. This paper shows that the selection order during labeling is crucial for search efficiency.

2 The Grammar School Timetabling Problem

In Germany the aim of visiting grammar school is to acquire college graduation. The German grammar school comprises nine grades, which are numbered from five to thirteen. The timetabling problem can vary greatly between different grammar schools. The Maria-Theresia-Gymnasium (MTG) at Augsburg, Germany is a grammar school with about 800 pupils, who are educated by about 60 teachers in three directions of study. Timetabling is carried out in three main steps. First, pupils are allocated into classes. Second, teachers are assigned. Third, timetables and room assignments are developed. Teachers are allocated in advance of the timetabling process, so the problem is to match up meetings of teachers with classes to particular time slots so that each particular teacher meets every class he or she is required to. Development of timetables is task of two dedicated teachers; the work is performed manually and takes about two weeks. Due to changes in structure, the problem of timetabling is to be solved from scratch every year.

This work deals with the 1997 instance of the MTG timetabling problem. The task consisted in creating timetables on base of a specification including classes, courses, assignment of teachers, course phase timetables, the timetables of sports education and the requirements to be met by timetables. In total, there were 967 lessons to place. The specification was compiled in cooperation with MTG. Since there is no lack of space at MTG, assignment of rooms played a minor role.

The problem was tackled and solved in part by means of CLP. For the timetabler CLP offers a variety of methods, mainly from operations research and artificial intelligence, developed to solve combinatorial problems efficiently [Pro93, Tsa93]. These methods require to declare the problem as a constraint satisfaction problem, i.e. a set of variables (representing lessons, for example), each associated with a domain of values it can take on, and a set of constraints among the variables. Constraints are relations which specify the space of solutions by forbidding combinations of values. In our application, we distinguish two kinds of constraints:

- *Resource constraints* make sure that rooms, equipment and teachers are available. In other words, resource constraints ensure physical feasibility. Reasons for unavailability of resources are mainly due to the competition of lessons for resources. Furthermore resources may be shared among different schools. In this case the times of unavailability are likely to be known in advance.
- *Quality constraints* make sure that the timetable is good or acceptable. Since preferences and priorities depend highly on school and grade, a precise definition of a good or acceptable timetable cannot be given in this broad context. In general a timetable will be considered good if it is in line with pedagogical principles, which improve the efficiency of education, and if it supports the working climate.

Most requirements can be expressed using the constraints `all_distinct` [Rég94] and `cumulative` [AB92]. Efficient implementations of these constraints are based on graph theory and operations research methods. The constraint

```
cumulative(Starts,Durations,Resources,Limit),
```

where the first three arguments are lists of equal length n of domain variables or integers, and `Limit` is a domain variable or an integer, can be thought of as constraining n tasks, each with a start time `Sj`, a duration `Dj`, and a resource amount `Rj`, such that the total resource consumption does not exceed `Limit` at any time. The constraint

```
all_distinct(Variables),
```

where `Variables` is a list of domain variables or integers, means that each variable is constrained to take a value that is unique among the variables.

3 Heuristics: A Guide to Search

Typically, constraint propagation is incomplete, and therefore has to be combined with search. Systematic search instantiates variables sequentially following some search order. If the procedure fails to extend a partial solution, decisions are undone and alternatives explored. Systematic search often relies on heuristics which define the order in which variables and values are chosen. Orders have been found to have a substantial effect on the efficiency of search [DM94]. A variety of heuristics defining orders have been proposed and examined in literature. Value selection has not attracted as much attention as variable selection [BR96]. Heuristics for variable selection often recommend to prefer variables which are most constrained [GMPW96]; this way search branches into one of the least constrained subproblems.

To solve our timetabling problem, we implemented several heuristics including the minimum remaining values heuristic [HE80], $e(n)$ [GMP$^+$96] and *prom* [Gee92]. The minimum remaining values heuristic dynamically orders variables by increasing cardinality of domains, i.e. it proposes to select one of the variables with the smallest domains with respect to the current state of computation. $e(n)$, a principle for variable selection proposed recently, branches into one of the subproblems which are expected to have a maximum number of solutions. *prom*, a general value selection heuristic, branches into one of the biggest subproblems. There is only one combination which leads to a solution: $e(n)$ together with *prom*. The resulting search procedure solved the problem within 15 minutes on a personal computer, which is a very reasonable period of time. The plan is physically feasible and satisfying with respect to the quality of education, but teachers' timetables are not acceptable because teachers' wishes and some quality constraints have not been considered.

4 Conclusion

This paper presented an approach to solving German grammar school timetabling using finite domain constraints and pointed out the prominent role of heuristics. In combination with *e(n)* for variable selection and *prom* for value selection this approach yields good results for this hard problem with many variables and an extremely large search space.

Our prototype can only handle hard constraints, i.e. conditions that must be satisfied. We plan to treat soft constraints, i.e. conditions that may be violated but should be fulfilled (e.g. teachers' wishes).

There are a number of techniques that have been used to solve the timetabling problem. In recent years, the idea of hybridizing constraint programming and operations research techniques has been tested and shown to improve upon either technology used alone. One interesting direction for future work is to develop combined methods for our timetabling problem that hopefully exploit the good qualities of the various methods.

References

[AB92] Abderrahmane Aggoun and Nicolas Beldiceanu. Extending CHIP in order to solve complex scheduling and placement problems. In *Actes des Journées Francophones de Programmation en Logique*, 1992.

[BR96] Christian Bessiere and Jean-Charles Régin. Mac and combined heuristics: Two reasons to forsake fc (and cbj?) on hard problems. In *Second International Conference on Principles and Practice of Constraint Programming*, LNCS 1118, pages 61–75. Springer, 1996.

[DM94] Rina Dechter and Itay Meiri. Experimental evaluation of preprocessing algorithms for constraint satisfaction problems. *Artificial Intelligence*, 68:211–241, 1994.

[FA97] Thom Frühwirth and Slim Abdennadher. *Constraint programming: Foundations and applications (In German: Constraint-Programmierung: Grundlagen und Anwendungen).* Springer, 1997.

[Gee92] Pieter A. Geelen. Dual viewpoint heuristics for binary constraint satisfaction problems. In Bernd Neumann, editor, *Proceedings of the 10th European Conference on Artificial Intelligence*, pages 31–35. John Wiley & Sons, 1992.

[GMP+96] Ian P. Gent, Ewan MacIntyre, Patrick Prosser, Barabara M. Smith, and Toby Walsh. An empirical study of dynamic variable ordering heuristics for the constraint satisfaction problem. In Eugene C. Freuder, editor, *Second International Conference on Principles and Practice of Constraint Programming*, pages 179–193. Springer, 1996.

[GMPW96] Ian P. Gent, Ewan MacIntyre, Patrick Prosser, and Toby Walsh. The constrainedness of search. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, pages 246–252. AAAI Press / MIT Press, 1996.

[HE80] Robert M. Haralick and G. L. Elliott. Increasing tree search efficiency for constraint satisfaction problems. *Artificial Intelligence*, 14:263–313, 1980.

[Int98] Intelligent Systems Laboratory, Swedish Institute of Computer Science. *SICStus Prolog User's Manual, release 3.7*, 1998.

[JM94] Joxan Jaffar and Michael J. Maher. Constraint logic programming: A survey. *The Journal of Logic Programming*, 19 & 20:503–582, 1994.

[MS98] Kim Marriott and Peter Stuckey. *Programming with Constraints: An Introduction*. The MIT Press, 1998.

[Pro93] Patrick Prosser. Hybrid algorithms for the constraint satisfaction problem. *Computational Intelligence*, 9(3):268–299, 1993.

[Rég94] Jean-Charles Régin. A filtering algorithm for constraints of difference in CSPs. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, volume 1, pages 362–367. AAAI Press / MIT Press, 1994.

[Tsa93] Edward Tsang. *Foundations of Constraint Satisfaction*. Academic Press, 1993.

[Wal96] Mark Wallace. Practical applications of constraint programming. *Constraints Journal*, 1:139–168, 1996.