Scheduling with Soft CLP(*FD*) Solver*

Tomáš Černý, Hana Rudová

Faculty of Informatics, Masaryk University
Botanická 68a, Brno 602 00, Czech Republic
hanka@fi.muni.cz

Timetabling problems often consists from various requirements which can not be satisfied together [11]. Unsatisfiable requirements can be handled within optimization criteria as *soft constraints*. Such soft constraints may not be satisfied if there are some contradictions. The remaining hard constraints must be still satisfied. The set of hard and soft constraints can be naturally expressed using constraint programming [3]. Unfortunately there is no system available which would allow to use both hard and soft constraints together. There are various systems implementing soft constraints [2] but none of them allows to combine efficient constraint propagation algorithms for classical constraint satisfaction problems together with the propagation for soft constraints.

Our implementation of the system for timetabling problem at Purdue University [9] was able to use both hard and soft constraints together. Hard constraints were available from SICStus Prolog CLP(*FD*) library [1] and soft constraints were implemented using the Soft CLP(*FD*) Solver [8]. This solver for soft constraints includes the specific set of soft constraints needed for the Purdue University timetabling problem. Our current intent is to generalize this proposal and implement the Soft CLP(*FD*) Solver as an open extendable library able to solve a wide class of problems.

The new solver allows to define soft constraints as it is shown in the course timetabling example in Figure 1. Hard constraint `disjoint2` ensures that all

```
(1) class_timetabling( TimesAndRooms ) :-
        ...
(2)     disjoint2( [class(Time, Duration, Room, 1) | Rest ] ),
(3)     serialized( Times, Durations ),
(4)     Time in 7..8 @ discouraged,
(5)     TimeForLecture #< TimeForSeminar @ preferred,
(6)     soft_serialized(TimeForSeminar1,TimeForSeminar2,TimeForSeminar3),
        ...
(7)     labeling( TimesAndRooms ).
```

Fig. 1. Course timetabling example

* This work is supported by the Ministry of Education of the Czech Republic under the research intent No. 0021622419.

E. K. Burke, H. Rudová (Eds.): PATAT 2006, pp. 381–383. ISBN 80-210-3726-1.

classes in the list do not overlap in time and space. Another hard constraint `serialized` allows for example to teach all classes of the same teacher at different times. The soft constraint `(4)` discourages placement of the class identified by its starting `Time` at seven or eight o'clock. Soft constraints can specify desired or undesired relations between classes `(5)`. Soft global constraint `soft_serialized` allows to express that seminars of the same course should be preferably taught at different times.

While the original solver is based on partial forward checking [4], the new proposal allows to consider AC* and NC* consistency [6]. The variables in soft constraints are called *preference variables* and they are implemented using the attributed variable [5]. The attribute of each preference variable stores the current cost for each value present in the domain of the variable. Each preference variable is also a standard domain variable which allows to include it in any (hard) constraint of the CLP(*FD*) library. The *lower bound* of the solution is represented as a domain variable. Potential violations of soft constraints contribute to this cost and backtracking occurs when the lower bound of the current partial solution is greater than some existing upper bound.

The current implementation contains the basic unary and binary soft constraints. We plan to implement some soft global constraints based on principles described in [7]. There are some proposals for soft global cardinality constraints [10] we would like to study. These could be very interesting for solving of the employee timetabling problems. We also intend to use the solver for machine scheduling problems where various soft constraints are specified by users submitting tasks to the problem.

References

1. Mats Carlsson, Greger Ottosson, and Björn Carlson. An open-ended finite domain constraint solver. In *Programming Languages: Implementations, Logics, and Programming*. Springer-Verlag LNCS 1292, 1997.
2. Simon de Givry. Soft constraint satisfaction problem, 2005. `http://carlit.toulouse.inra.fr/cgi-bin/awki.cgi/`.
3. Rina Dechter. *Constraint Processing*. Morgan Kaufmann Publishers, 2003.
4. Eugene C. Freuder and Richard J. Wallace. Partial constraint satisfaction. *Artificial Intelligence*, 58:21–70, 1992.
5. Christian Holzbaur. *Specification of Constraint Based Inference Mechanism through Extended Unification*. PhD thesis, University of Vienna, 1990.
6. Javier Larrosa and Thomas Schiex. Solving weighted CSP by maintaining arc consistency. *Artificial Intelligence*, 159(1–2):1–26, 2004.
7. T. Petit, J.-C. Régin, and C. Bessiere. Meta-constraints on violations for over constrained problems. In *12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI00)*, pages 358–365.
8. Hana Rudová. Soft CLP(*FD*). In Susan Haller and Ingrid Russell, editors, *Proceedings of the 16th International Florida Artificial Intelligence Symposium, FLAIRS-03*, pages 202–206. AAAI Press, 2003.

9. Hana Rudová and Keith Murray. University course timetabling with soft constraints. In Edmund Burke and Patrick De Causmaecker, editors, *Practice and Theory of Automated Timetabling, Selected Revised Papers*, pages 310–328. Springer-Verlag LNCS 2740, 2003.
10. Willem Jan van Hoeve, Gilles Pesant, and Luois-Martin Rousseau. On global warming (softening global constraints). In *6th International Workshop on Preferences and Soft Constraints*, 2004.
11. Anthony Wren. Scheduling, timetabling and rostering – a special relationship? In Edmund Burke and Peter Ross, editors, *Practice and Theory of Automated Timetabling*, pages 46–75. Springer-Verlag LNCS 1153, 1996.