The Aimms Interface to Constraint Programming

Willem-Jan van Hoeve[1], Marcel Hunting[2], and Chris Kuip[2]

[1] Tepper School of Business, Carnegie Mellon University
[2] Paragon Decision Technology

Abstract. We present an extension of the modeling system Aimms to handle constraint programming problems. Our goal is to provide a more accessible interface to CP technology than current systems offer. We first present basic CP modeling constructs that can be realized with minimum changes to the existing syntax. We then discuss the handling of global constraints. Lastly, we present our extensions to modeling scheduling problems, based on the now-classical representation as activities and resources. An important benefit of the Aimms interface to CP is the ease with which hybrid CP/OR solution methods can be developed.

1 Introduction

From its inception, the academic field of Constraint Programming (CP) has been coupled with well-engineered solvers that were applied to challenging, often large-scale, industrial problems (e.g., CHIP, Cosytec, ILOG Solver, SICStus, Eclipse, Choco, Kalis, Gecode, Comet). Most, if not all, of these solvers require a considerable amount of training before they can be used by an engineering student or Operations Research practitioner, let alone an MBA student. One reason for this is that each of these solvers has its own specific modeling language, for example in the style of Prolog or C++. Another reason is that CP requires a different modeling style than mathematical programming, to the extent that even a dynamic search strategy can be modeled in most systems. Finally, even though most available solvers share the most important characteristics of CP, each comes with its own library of global constraints, associated filtering algorithms, modeling concepts (e.g., resource constrained scheduling, set variables, local search), and specialized search techniques. Consequently, a problem may be modeled and solved in entirely different ways by different solvers. All of this limits the accessibility of CP to typical Operations Research practitioners.

One of the main goals of the project described in this abstract is to make CP more accessible to a wider range of practitioners, from industrial OR experts to MBA students. In order to achieve this, we have developed a Constraint Programming extension of the modeling system Aimms, keeping the following requirements and targets in mind. First, we would like the difference between standard mathematical programming models and CP models to be minimal for a user, and in particular for an Aimms user. Second, we must offer the most

powerful CP technology that is available (e.g., access to advanced algorithms for resource-constrained scheduling), as well as all common global constraints. Third, the system should be designed for solving hard industrial problems. Note that these targets can be conflicting, e.g., the support of global constraints for their filtering power does not align with standard mathematical programming terminology.

2 Background and Related Work

Several other industrial and academic modeling languages/systems have been developed to make the use of CP and hybrid methods more accessible. Academic systems focusing on the integration of solvers include Zinc [9] and SIMPL [13]. Industrial systems include IBM ILOG CPLEX Optimization Studio (with the OPL language [11]), Fico Xpress Optimization Studio (with the Mosel language [4]), and Comet [12]. Even though these systems are powerful and provide advanced interfaces, they mostly appeal to specialized developers, and suffer from the aforementioned shortcomings with respect to accessibility to non-specialists, in our opinion. Our goal is to contribute to these developments by focusing specifically at a solver-independent, intuitive graphical modeling interface, as offered by the Aimms system, in order to attract non-experts to use CP technology.

The modeling language underlying Aimms [2,10] was originally developed in a similar spirit as GAMS [3] or AMPL [6]. Similar to those systems, it is based on an algebraic syntax and offers access to (at least) ILP, QP, and NLP technology. The main difference with these other languages, however, is that model development in Aimms revolves around a graphical modeling interface depicting the hierarchical structure in a model formulation. This enables the user to formulate a problem in an intuitive and naturally decomposed manner. A second important feature of Aimms is that it offers user-developed 'pages', that can be used to graphically depict solutions and even to build entire end-user applications that can be deployed in practice.

3 Contributions

The Aimms extension to CP supports all common global constraints, ranging from `AllDifferent` to `Sequence` and `BinPacking`. In addition, it supports advanced scheduling concepts based on the classical representation using 'activities' and 'resources' [1], as well as specialized global scheduling constraints as introduced recently in [8]. Interestingly, many modeling concepts from CP were already present in the existing Aimms syntax, albeit restricted to non-variable identifiers. Namely, Aimms already offers all standard arithmetic, logical, and set related operators. This allowed us to provide CP functionality with minimal changes to the existing syntax in many cases. Finally, we have built an interface to the CP solvers IBM ILOG CP Optimizer and Gecode. We refer to [7] for more details.

In summary, the most important contribution of our extension to the practice of CP is that less-experienced users can access CP technology in a more intuitive way, and furthermore make use of all standard features of AIMMS including its advanced graphical interface. In addition, we remark that the related modeling systems GAMS and AMPL do not support CP technology,[1] while OPL, Mosel, and Comet do not provide access to NLP solvers such as CONOPT, KNITRO, IPOPT and LGO. With the addition of our CP interface, AIMMS is currently the only industrial-strength optimization modeling system that provides access to LP, MIP, QP, NLP, and CP solvers, which makes it possible to build integrated models combining all these technologies.

References

1. P. Baptiste, C. Le Pape, and W. Nuijten. *Constraint-Based Scheduling: Applying Constraint Programming to Scheduling Problems*. Kluwer Academic Publishers, 2001.
2. J. Bisschop and R. Entriken. *AIMMS: The modeling system*. Paragon Decision Technology, 1993.
3. J. Bisschop and A. Meeraus. On the development of a general algebraic modeling system in a strategic planning environment. *Mathematical Programming Study*, pages 1–29, 1982.
4. Y. Colombani and S. Heipcke. Mosel: An Extensible Environment for Modeling and Programming Solutions. In *Proceedings of the Fourth International Workshop on Integration of AI and OR techniques in Constraint Programming for Combinatorial Optimisation Problems (CPAIOR)*, 2002.
5. R. Fourer and D.M. Gay. Extending an algebraic modeling language to support constraint programming. *INFORMS Journal on Computing*, 14:322–344, 2002.
6. R. Fourer, D.M. Gay, and B.W. Kernighan. AMPL: A Modeling Language for Mathematical Programming. *Management Science*, 36:519–554, 1990.
7. W.J. van Hoeve, M. Hunting, and C. Kuip. Constraint programming. In *AIMMS 3.12: The Language Reference*, chapter 21. Paragon Decision Technology, 2011.
8. P. Laborie. IBM ILOG CP Optimizer for Detailed Scheduling Illustrated on Three Problems. In *Proceedings of CPAIOR*, volume 5547 of *LNCS*, pages 148–162. Springer, 2009.
9. K. Marriott, N. Nethercote, R. Rafeh, P.J. Stuckey, M. Garcia De La Banda, and M. Wallace. The design of the Zinc modelling language. *Constraints*, 13(3):229–267, 2008.
10. M. Roelofs and J.J. Bisschop. *AIMMS 3.11: The Language Reference*. Paragon Decision Technology, 2010. http://www.aimms.com/downloads/manuals/language-reference.
11. P. Van Hentenryck. *The OPL Optimization Programming Language*. The MIT Press, 1999. With contributions by I. Lustig, L. Michel, and J.-F. Puget.
12. P. Van Hentenryck and L. Michel. *Constraint-Based Local Search*. The MIT Press, 2005.
13. T. Yunes, I.D. Aron, and J.N. Hooker. An integrated solver for optimization problems. *Operations Research*, 58(2):342–356, 2010.

[1] We note that the AMPL language extensions to CP, as described in [5], have not yet been implemented.