Branch-and-Price Solving in G12

Jakob Puchinger[1], Peter J. Stuckey[2], Mark Wallace[3], and Sebastian Brand[2]

[1] Austrian Institute of Technology
Vienna, Austria
jakob.puchinger@ait.ac.at

[2] NICTA Victoria Research Laboratory
Department of Computer Science & Software Engineering
University of Melbourne, Australia
{pjs,sbrand}@csse.unimelb.edu.au

[3] School of Computer Science and Software Engineering
Monash University, Melbourne, Australia
mgw@mail.csse.monash.edu.au

1 Introduction

Combinatorial optimisation problems are easy to state but hard to solve, and they arise in a huge variety of applications. Branch-and-price is one of many powerful methods for solving them. This paper describes how Dantzig-Wolfe decomposition, column generation and branch-and-price are integrated into the hybrid optimisation platform G12 [13]. The G12 project is developing a software environment for stating and solving combinatorial problems by mapping a high-level model of the problem to an efficient combination of solving methods.

The G12 platform consists of three major components, the modelling language Zinc [5], the model transformation language Cadmium [3], and several internal and external solvers written and/or interfaced using the general-purpose programming language Mercury [12]. All solvers and solver instances are specified in terms of their specific capabilities, i.e. the type of problems they can solve, the type of information they can return, and how they solve a problem. The branch-and-price solving in G12 was first described in detail in [9].

The practical usefulness of column generation and branch-and-price has been well-established over the last 20 years [2, 1]. More recently it has emerged that column generation provides an ideal method for combining approaches, such as constraint programming, local search, and integer/linear programming [7, 11, 8].

Systems such as ABACUS [6] and COIN/BCP [10] and others offer facilities to support the implementation of branch-and-price. However, these systems require the user to understand the technical details of branch-and-price, supporting algorithm implementation rather than problem modelling. The first attempt to provide a column generation library was in ECL^{i}PS^e [4]. This system introduced the idea of an aggregate variable appearing in the master problem to represent a set of values returned as columns from multiple solutions to identical subproblems. However this library assumes a fixed set of variables in each subproblem, and precludes search choices which break some of the subproblem symmetries.

Dagstuhl Seminar Proceedings 09261
Models and Algorithms for Optimization in Logistics
http://drops.dagstuhl.de/opus/volltexte/2009/2164

2 G12 Branch-and-Price Solving

We present the different aspects of G12 branch-and-price solving using three examples. All of these examples are accompanied by extensive computational experiments showing the effectiveness of our system.

The first example, a trucking problem, is used to show how high-level models are mapped to a standard column generation approach with branch-and-price on the original variables. In order to use Dantzig-Wolfe decomposition and column generation on a high-level model in G12, annotations describing what parts define the sub-problems, which solver is to be used for each subproblem, and which solver is to be used for the master problem are introduced. The annotation of the original model is done by the user. In a second step an automatic CADMIUM transformation is applied. It performs a Dantzig-Wolfe decomposition on the model and separates original, master and subproblem variables. It further adds constraints linking these variables, informing the underlying column generation module about the specific structure of the model. The model is then solved by column generation and branching on the original variables.

The second example, the cutting stock problem, is used to demonstrate the variable aggregation facilities of the system. The main purpose of variable aggregation is to avoid problem symmetries and thus significantly reduce the required search effort. The variable aggregation is controlled by annotations specified by the user, informing the underlying system which subproblems should be aggregated. A CADMIUM transformation is then applied to create an aggregated version of the variables and constraints. The column generation module manages the disaggregation of variables required in the branching phase, so that branching on the original problem variables is still possible.

In the third example, the two-dimensional bin packing problem, the use of specialised branching rules on subproblem variables is shown. Such branching rules are very effective in reducing symmetry, since they do not require variable disaggregation. However, in contrast to branching on original variables, they introduce modifications of the subproblems. In the current system the branching rules have to be developed at the MERCURY level, but they are selected at the modelling level using annotations.

3 Conclusion

The presented system allows one to specify column generation and branch-and-price on high-level models without the need to implement the technical details. This allows users to experiment with different variants of this specific way of hybrid solving and to compare it to other solver mappings in G12.

One interesting challenge arising out of this work is how to automatically detect identical subproblems. This is a completely novel form of automated symmetry detection, which is of significant practical value.

Acknowledgements

We would like to thank the members of the G12 team at NICTA VRL for helpful discussions and implementation work.

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

1. C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329, 1998.
2. G. Desaulniers, J. Desrosiers, and M. Solomon, editors. *Column Generation.* GERAD 25th Anniversary Series. Springer, 2005.
3. G. J. Duck, P. J. Stuckey, and S. Brand. ACD term rewriting. In S. Etalle and M. Truszczynski, editors, *Logic Programming (ICLP 2006)*, volume 4079 of *LNCS*, pages 117–131. Springer, 2006.
4. A. Eremin. *Using Dual Values to Integrate Row and Column Generation into Constraint Logic Programming.* PhD thesis, Imperial College London, 2003.
5. M. J. Garcia de la Banda, K. Marriott, R. Rafeh, and M. Wallace. The modelling language Zinc. In F. Benhamou, editor, *Principles and Practice of Constraint Programming (CP'06)*, volume 4204 of *LNCS*, pages 700–705. Springer, 2006.
6. M. Jünger and S. Thienel. The ABACUS system for branch-and-cut-and-price algorithms in integer programming and combinatorial optimization. *Software: Practice and Experience*, 30(11):1325–1352, 2000.
7. U. Junker, S. E. Karisch, N. Kohl, B. Vaaben, T. Fahle, and M. Sellmann. A framework for constraint programming based column generation. In J. Jaffar, editor, *Principles and Practice of Constraint Programming (CP'99)*, volume 1713 of *LNCS*, pages 261–274. Springer, 1999.
8. J. Puchinger and G. R. Raidl. Models and algorithms for three-stage two-dimensional bin packing. *European Journal of Operational Research*, 183(3):1304–1327, 2007.
9. J. Puchinger, P. J. Stuckey, M.Wallace, and S. Brand. From high-level model to branch-and-price solution in G12. In L. Perron and M. A. Trick, editors, *CPAIOR 2008*, volume 5015 of *LNCS*, pages 218–232. Springer, 2008.
10. T. Ralphs and L. Ladanyi. COIN/BCP user's manual, 2001.
11. L.-M. Rousseau, M. Gendreau, G. Pesant, and F. Focacci. Solving VRPTWs with constraint programming based column generation. *Annals of Operations Research*, 130(1):199–216, 2004.
12. Z. Somogyi, F. Henderson, and T. Conway. The execution algorithm of Mercury, an efficient purely declarative logic programming language. *Journal of Logic Programming*, 29(1-3):17–64, 1996.
13. P. J. Stuckey, M. J. G. de la Banda, M. J. Maher, K. Marriott, J. K. Slaney, Z. Somogyi, M. Wallace, and T. Walsh. The G12 project: Mapping solver independent models to efficient solutions. In P. van Beek, editor, *Principles and Practice of Constraint Programming (CP'05)*, volume 3709 of *LNCS*, pages 13–16. Springer, 2005.