Integer Programming Models in AI Planning: Preliminary Experimental Results

Thomas Vossen **Michael Ball**
Robert H. Smith School of Business
and Institute for Systems Research
University of Maryland
College Park, MD 20742 USA
{tvossen, mball}@rhsmith.umd.edu

Amnon Lotem **Dana Nau**
Department of Computer Science
and Institute for Systems Research
University of Maryland
College Park, MD 20742 USA
{lotem, nau}@cs.umd.edu

Introduction

As one of the challenges posed in their paper "Ten Challenges in Propositional Reasoning and Search," Selman *et al.* (1997) mention the development of Integer Programming (IP) models and methods for propositional reasoning. Even though it is straightforward to formulate a satisfiability problem as an integer programming model, their concern is that the basic technique used to solve integer programs—that is, the linear (LP) relaxation of the problem—does not guide the selection of values in solving the the integer program. As a reason for this, they mention that the LP relaxation of the problem usually sets most values to $\frac{1}{2}$.

Recently, we have begun to investigate the use of IP methods in the planning domain. Even though we are still in the early stages of our research, our preliminary experiments strongly suggest that IP methods do have something to offer for these problems. In particular, it appears that the LP relaxation of the problem *does* provide guidance in solving the IP. Below we briefly summarize the IP formulations and the results of our experiments.

Integer Programming Formulations

The class of planning problems we considered was based on the STRIPS paradigm. STRIPS planning problem instances were formulated as integer programs using the SATPLAN encodings proposed by Kautz and Selman (1996). In these encodings, a planning problem is converted to a satisfiability problem. Using largely straightforward techniques (Hooker 1988; Blair, Jeroslow, and Wang 1986) for converting clauses to linear constraints, we formulated the resulting SAT-problem as an integer program. The types of SATPLAN encodings we considered were the following:

- Linear Encoding with Explanatory Frame Axioms (Kautz and Selman 1996);
- Lifted Causal Encoding (Kautz, McAllester, and Selman 1996).

In order to obtain a more concise formulation, exclusiveness constraints were modeled slightly differently. This was done introducing constraints which state that the sum of mutually exclusive actions equals 1. For instance, the fact that exactly one action $a \in A$ occurs at each time instant t is modeled as

$$\sum_{a \in A} a_t = 1 \qquad \forall t.$$

Another modification we made was that we didn't restrict all variables 0-1 integers. In the linear encodings for instance, the integrality of all fluents f was relaxed to be $0 \leq f \leq 1$. The reason for this is that the integrality of these variables is implied by the integrality of the action variables. Similarly, in the causal encoding only the variables representing links were restricted to be integer. As a consequence of doing this, none of the relaxed variables will be selected in the branch and bound tree.

For the lifted encodings, the objective was set to maximize the sum of the goal state fluents (i.e. to maximize the number of goal met). However, as we also included the constraints that goal states are fulfilled at the final time t instant, this didn't carry any information. For the causal encoding, we set the objective to minimize the sum of the 'before' variable (i.e. to obtain a "minimum" partial order).

Preliminary Experimental Results

The IP formulations mentioned above were tested on the blocks world problem. The number of blocks were varied from 3 to 9, and the number of moves required from 3 to 7. All instances have multiple stacks of blocks. The integer programs were solved using Cplex 4.0, using a SunSparc 5 workstation. The results are shown in the following tables. In all tables, nodes represents the number of nodes visited in the branch and bound procedure, and iterations the number of simplex iterations performed. Times are in seconds.

While the computation times for both formulations is generally not impressive, the number of nodes required is generally very low. This suggests that the LP relaxation is strong and that it provides strong guidance in the selection of variables to branch on. In fact, for instance BW large A (Kautz and Selman 1996), the LP relaxation itself finds an integer solution! For larger instances however, the size of the LP formulation and

Table 1: Linear encoding, explanatory frame axioms.

blocks	moves	nodes	iterations	time	comment
3	3	2	61	0.07	anomaly
4	4	1	393	1.37	
5	5	4	1507	10.80	
7	7	13	9292	779.20	
9	6	0	31639	1705.70	BW large A

Table 2: Causal link planner. Reports are for the first integer solutions found.

blocks	moves	nodes	iterations	time	comment
3	3	3	414	0.81	anomaly
4	4	2	1168	6.41	
5	5	28	5641	74.30	
7	7	20	14266	535.60	
9	6	2	5756	198.20	BW large A

correspondingly the time needed to solve the LP relaxation become too large.

Conclusions

Although Selman *et al.* (1997) reported difficulty in making effective use of IP techniques for propositional reasoning in general, our preliminary results suggest that IP techniques may potentially work well for AI planning problems. The fact that the test problems have been solved with a very small number of nodes indicates that the linear programming relaxation is strong and that reduced computation times can be achieved by engineering the LP solution process. We have not yet tried to tune the LP solver for use on AI planning problems, but these problems appear to be excellent candidates for constraint and column generation techniques, which should allow for the solution of larger problems.

One reason why we find this prospect exciting is that the IP formulation represents the planning domain as a set of numeric equations, which should make it quite easy to incorporate numeric constraints and computations into formulations of planning domains. The ability to do this is a critical need for real-world planning, and it is not addressed adequately in most existing AI planning systems (Nau *et al.* 1998).

Our work is still in its early stages, and the results reported above are still preliminary. Although the first results are encouraging, our first goal is to investigate whether they generalize to other planning problems (Logistics problem, Rocket problem). Also, we are looking at alternative formulations, such as the graphplan and SAT encodings (Kautz, McAllester, and Selman 1996). Further ahead, we might want to investigate the use of constraint and column generation techniques, in order to speed up the LP relaxation.

Acknowledgements

This work has been supported in part by NSF grants EEC 94-02384 and IRI-9306580, and ARPA grant DABT-95-C0037. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funders.

References

Blair, C.E., Jeroslow, R.G., and J.K. Lowe. 1986. Some results and experiments in programming techniques for propositional reasoning. *Computers and Operations Research* 13:633–645.

A. L. Blum and M. L. Furst. 1997. Fast Planning Through Planning Graph Analysis. *Artificial Intelligence*, 90(1–2):281–300.

J. N. Hooker. 1988. A quantative approach to logical inference. *Decision Support Systems* 4:45–69.

Henry Kautz, David McAllester, and Bart Selman. 1996. Encoding plans in propositional logic. *Proc. KR-96.*

Henry Kautz and Bart Selman. 1996. Pushing the envelope: Planning, propositional logic, and stochastic search. *Proc. AAAI-96.*

D. S. Nau, S. J. Smith and Kutluhan Erol. 1998. Control strategies in HTN planning: theory versus practice. In *Proc. IAAI-98*, to appear.

B. Selman, H. Kautz, and D. McAllester. 1997. Ten challenges in propositional reasoning and search. In *Proc. Fifteenth International Joint Conf. Artificial Intelligence (IJCAI-97)*, Nagoya, Japan.