Planning with Numerical State Variables through Mixed Integer Programming

Menkes van den Briel
Department of Industrial Engineering
Arizona State University
Tempe AZ, 85287-8809
menkes@asu.edu

Subbarao Kambhampati
Department of Computer Science
Arizona State University
Tempe AZ, 85287-8809
rao@asu.edu

Thomas Vossen
Leeds School of Business
University of Colorado at Boulder
Boulder CO, 80309-0419
vossen@colorado.edu

Abstract

We extend the state-of-the-art IP formulations for classical planning to include resources and optimization objectives. We present our initial findings and show some preliminary results.

Introduction

One of the most compelling reason for using integer programming (IP) and mixed integer programming (MIP) techniques in planning is when the planning problem contains numerical state variables. Numerical state variables appear in many practical planning domains and are often accompanied by linear numerical constraints and optimization criteria, which are naturally supported by the IP framework. Traditionally, IP has been used to tackle hard combinatorial optimization problems that arise in the field of operation research. However, recent work has shown that IP techniques also show great potential in their ability to solve classical AI planning problems and can compete with the most efficient SAT-based encodings (van den Briel, Vossen, & Kambhampati 2005).

Currently, we are investigating the use of IP techniques in numerical planning by extending the state-of-the-art IP formulations for classical planning and by adding on the work of Kautz and Walser (1999). We will exploit the strength of IP techniques to solve optimization problems with numerical constraints, to extend to AI planning problems that involve numerical state variables and numerical constraints. Even though we are still in the early stages of our research, our first observations and preliminary results suggest that we can improve previous IP approaches to solve these type of planning problems more effetively. Below we will give a brief summary of our IP formulations and show some initial results.

Numerical State Variables

We often refer to numerical state variables as *resources*. Heuristics for planning with resources have been studied by several different works (Do & Kambhampati 2001; Halsum & Geffner 2001; Hoffmann 2002; Refandis & Vlahavas 2000). Studies on IP formulations for resource planning, however, are not as plentiful. Wolfman and Weld (1999) use LP formulations in combination with a satisfiability-based planner to solve resource planning problems, and Kautz and Walser (1999) use IP formulations for resource planning problems that incorporate action costs and complex objectives.

In order to reason about resources, actions are extended to include resource preconditions and effects. Koehler (1998) provides a general framework in which a resource precondition is a simple linear inequality that must hold in each state where the action is applicable, and the action effects are to produce (increase), consume (decrease), or provide (assign) the value of a resource. A resource is called *reusable* if it can only be borrowed, that is, it cannot be consumed or produced by any action. In all other cases a resource is called *consumable*. A reusable resource is *sharable* if it can be borrowed by more than one action at the same time, otherwise it is non-sharable.

The AIPS-2002 planning competition introduced several planning domains with resources. The language that was used in this competition, PDDL2.1 (Fox & Long 2003), incorporates the possibility to define numerical constraints and effects on numerical state variables. Table 1 gives all the numeric domains of this competition and lists all the resource variables. If there exists an action in the domain that has an effect on a resource variable, then that resource variable is listed in the corresponding action effect column. In addition, a type and bounds (where C is some constant) on the resource are given.

We say a resource is *monotonic* if it can only be produced (monotonic$^+$), or if it can only be consumed (monotonic$^-$). We say a resource is *nonmonotonic* if it can both be produced and consumed (nonmonotonic), and if it can be provided (nonmonotonic$^=$). These resource types are used to categorize the resource constraints in our IP formulations.

Integer Programming Formulations

Numerical constraints such as $\sum_{j \in B} a_j x_j + \sum_{j \in C} g_j y_j \leq b$, where a_j and g_j are real numbers, B the set of binary variables, and C the set

Domain	Increase	Decrease	Assign	Type	Bounds
Depots	(current-load ?z)	(current-load ?z)		nonmonotonic	$[0, C]$
	(fuel-cost)			monotonic$^+$	$[0, \infty)$
Driverlog	(driven)			monotonic$^+$	$[0, \infty)$
	(walked)			monotonic$^+$	$[0, \infty)$
Rovers	(energy ?x)	(energy ?x)		nonmonotonic	$[0, \infty)$
	(recharges)			monotonic$^+$	$[0, \infty)$
Satellite	(fuel-used)			monotonic$^+$	$[0, \infty)$
	(data-stored)			monotonic$^+$	$[0, \infty)$
		(fuel ?s)		monotonic$^-$	$[0, C]$
		(data-capacity ?s)		monotonic$^-$	$[0, C]$
Settlers	(available ?r ?v)	(available ?r ?v)	(available ?r ?v)	nonmonotonic$^=$	$[0, \infty)$
	(available ?r ?p)	(available ?r ?p)		nonmonotonic	$[0, \infty)$
	(space-in ?v)	(space-in ?v)		nonmonotonic	$[0, \infty)$
	(labor)			monotonic$^+$	$[0, \infty)$
	(pollution)			monotonic$^+$	$[0, \infty)$
UMT	(weight-load-v ?v)	(weight-load-v ?v)		nonmonotonic	$[0, \infty)$
	(volume-load-v ?v)	(volume-load-v ?v)		nonmonotonic	$[0, \infty)$
	(volume-load-l ?l)	(volume-load-l ?l)		nonmonotonic	$[0, \infty)$
Zenotravel	(onboard ?a)	(onboard ?a)		nonmonotonic	$[0, \infty)$
	(total-fuel-used)			monotonic$^+$	$[0, \infty)$
		(fuel ?a)	(fuel ?a)	nonmonotonic$^=$	$[0, C]$

Table 1: The numeric domains of the AIPS-2002 planning competition

of continuous and integer variables, have received a great deal of attention in the field of mixed integer programming (Savelsbergh 1994). We integrate some of the ideas presented in this field to deal with the numerical constraints and variables that are present in resource planning domains.

Our IP formulations for resource planning are an extension to the IP formulations given by (van den Briel, Vossen, & Kambhampati 2005). In this presentation, we will limit our focus on dealing with the numerical state variables, the propositional variables are dealt with in the same way as in van den Briel, Vossen and Kambhampati (2005). That is, propositional variables are transformed into multi-valued state variables, and changes in the state variables are modeled as flows in an appropriately defined network. As a consequence, the resulting IP formulations can be interpreted as a network flow problem with additional side constraints.

We will use the following notation:

- A: the set of ground actions
- R: the set of resources
- T: the maximum number of plan steps
- $prod(a)$, $cons(a)$, $prov(a)$: the set of resources that appear respectively as produce, consume, provide effects for action $a \in A$
- $produce_{a,r}$, $consume_{a,r}$, $provide_{a,r}$: the amount of resource $r \in R$ that is respectively produced, consumed, provided by action $a \in A$

In our formulations we use actions and numerical state variables, which we define as follows:

- $x_{a,t} \in \{0, 1\}$, for $a \in A, 1 \leq t \leq T$; $x_{a,t}$ is equal to 1 if action a is executed at plan step t, and 0 otherwise.
- $z_{r,t} \geq 0$, for $r \in R, 1 \leq t \leq T$; $z_{r,t}$ represents the value of resource r at plan step t. $z_{r,t}$ can be real or integer-valued and may be bounded from above. For now, we will assume that that each resource has a lower bound that can be normalized to 0.

Numerical state variables add constraints to the planning problem and they may appear in the optimization criteria of the planning problem. Next, we will discuss what constraints need to be added to the IP formulation in order to model the different resources.

Monotonic resources

Resources that behave monotonically can be modeled without introducing numerical state variables to the IP formulation. We can simply deal with these resources by adding them implicitly to the model. Let R^{mon+} and R^{mon-} be the set of resources of type monotonic$^+$ and monotonic$^-$ respectively. If the optimization criteria requires a monotonic resource to be minimized then we can simply setup the following objective function:

$$MIN \sum_{a \in A, 1 \leq t \leq T, r \in R^{mon+}: r \in prod(a)} produce_{a,r} x_{a,t} + \sum_{a \in A, 1 \leq t \leq T, r \in R^{mon-}: r \in cons(a)} consume_{a,r} x_{a,t}$$

Instead of representing monotonic resources by numerical state variables, we can simply deal with them

by directly working on the action effects. In case a monotonic resource is bounded then we add an extra constraint to satisfy this bound. For every monotonic$^+$ resource r with an upper bound U_r we must add the following constraint:

$$\sum_{a \in A, 1 \leq t \leq T, r \in R^{mon+}} produce_{a,r} x_{a,t} \leq U_r$$

Similarly, for every monotonic$^-$ resource r with a lower bound L_r and an initial value I_r we must add the following constraint:

$$\sum_{a \in A, 1 \leq t \leq T, r \in R^{mon-}} consume_{a,r} x_{a,t} \leq I_r - L_r$$

Note that in numeric planning domains where all resources are unbounded and monotonic, like the AIPS-2002 numeric driverlog domain, resource planning reduces to classical planning with cost sensitive actions.

Nonmonotonic resources

Resources that are nonmonotonic require the use of numerical state variables in the IP formulation. Since actions may increase or decrease the value of the resource, we need to keep track of their value over time. Let R^{non} be the set of nonmonotonic resources only affected by produce and consume effects of actions, and let $R^{non=}$ be the set of nonmonotonic resources that are affected by provide effects of actions. If nonmonotonic resources are to be minimized then we can add them to the objective function as follows:

$$MIN \sum_{a \in A, 1 \leq t \leq T, r \in R^{non} \cup R^{non=}: r \in prod(a)} produce_{a,r} x_{a,t} + \sum_{a \in A, 1 \leq t \leq T, r \in R^{non} \cup R^{non=}: r \in cons(a)} consume_{a,r} x_{a,t}$$

Also for each nonmonotonic resource $r \in R^{non}$ we must keep track of its value, hence we have the constraint:

$$z_{r,t-1} + \sum_{a \in A, r \in R^{non}: r \in prod(a)} produce_{a,r} x_{a,t} = \sum_{a \in A, r \in R^{non}: r \in cons(a)} consume_{a,r} x_{a,t} + z_{r,t}$$

When an action has a provide effect on a resource $r \in R^{non=}$ then the constraints for keeping track of the resource is more involved. In this case we currently use the linear inequalities as described by Kautz and Walser (1999).

Preliminary Results

For our preliminary studies of the effectiveness of our approaches we compare to the work of Kautz and Walser (KW). We use the airplane domain, which is a modified version of the airplane example from Penberthy and Weld (1994) and Koehler (1998). One or more airplanes can fly between a number of different airports. A fly action consumes fuel and a refuel action provides fuel, hence fuel is a nonmonotonic$^=$ resource. The goal is to take each passenger to his or her destination while minimizing the total fuel consumption. We setup an IP formulation, which we call 1SC, where the propositional variables are modeled as in van den Briel, Vossen, and Kambhampati (2005) and the resource is modeled as in Kautz and Walser (1999). Hence, the main difference in these two approaches is how the propositional variables are modeled. Also, the KW formulation is specifically modeled for this domain with one airplane, whereas our 1SC formulation is domain independent.

		1SC		KW	
Problem	Obj	Nodes	Time	Nodes	Time
(1, 2, 3)	3600	0	0.01	0	0.02
(1, 3, 3)	6780	0	0.02	0	0.04
(1, 6, 3)	9762	15	0.38	56	0.40
(2, 4, 3)	4500	0	0.05		
(2, 5, 3)	5644	42	0.10		
(2, 4, 4)	3939	4	0.14		
(2, 5, 4)	5014	2	0.19		
(2, 6, 4)	9273	606	0.28		
(3, 6, 5)	8914	292	0.49		
(3, 7, 5)	14919	2195	0.71		
(3, 8, 5)	21164	717	0.96		

Table 2: Results for the airplane domain, where the problem number is given by (#airplanes, #passengers, #cities).

Some results for the airplane domain are given in Table 2. All problems were solved to optimality, that is, the objective value represents the minimum amount of fuel needed to transport the passengers. Comparative analysis was made difficult by the fact that the KW formulation only works for single airplane scenarios, while our 1SC formulation can handle multiple airplanes and can be applied to a wide variety of other numerical planning domains. Nevertheless, on the problems that both approaches were able to solve our formulation outperformed the KW formulation in terms of number of branch-and-bound nodes and solution time.

Future Work

Integer programming provides a strong framework for dealing with numerical constraints and optimization criteria. So far, only a few researchers have looked into the application of IP techniques in planning with resources, and we believe that there is significant room for improvement. Even though we are still in the early stages of our research, there are some potential cutting planes that we may be able to detect and add to our IP formulations. For example, the bound constraints on

the monotonic resource variables can be interpreted as 0-1 knapsack constraints, and so, we could find knapsack cover inequalities. The constraints on the non-monotonic resources look very similar to constraints we see in lot-sizing problems (Pochet & Wolsey 1995), and so, we could find flow cover inequalities.

Besides adding cutting planes to our IP formulations, we are also looking at different ways to generalize the notion of action parallelism in planning domains that involve resources.

References

Do, M., and Kambhampati, S. 2001. Sapa: A domain-independent heuristic metric temporal planner. In *Proceedings of the European Conference on Planning (ECP-01)*, 109–120.

Fox, M., and Long, D. 2003. PDDL2.1: An extension of PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research* 20.

Halsum, P., and Geffner, H. 2001. Heuristic planning with time and resources. In *Proceedings of the Sixth European Conference on Planning (ECP-01)*, 121–132.

Hoffmann, J. 2002. Extending FF to numerical state variables. In *Proceedings of the 15th European Conference on Artificial Intelligence (ECAI-02*, 571–575.

Kautz, H., and Walser, J. 1999. State-space planning by integer optimization. In *AAAI-99/IAAI-99 Proceedings*, 526–533.

Koehler, J. 1998. Planning under resource constraints. In *Proceedings of the Thirteenth European Conference on Artificial Intelligence (ECAI-98)*, 489–493.

Penberthy, J., and Weld, D. 1994. Temporal planning with continuous change. In *Proceedings of the 12th National Conference on Artificial Intelligence*, 1010–1015.

Pochet, Y., and Wolsey, L. 1995. *Combinatorial Optimization: Papers from the DIMACS Special Year*, volume 20 of *DIMACS Series in Discrete Mathematics and Computer Science*. American Mathematical Society. chapter Algorithms and reformulations for lot-sizing problems, 245–294.

Refandis, I., and Vlahavas, I. 2000. Heuristic planning with resources. In *Proceedings of the Fourteenth European Conference on Artificial Intelligence (ECAI-00)*, 521–525.

Savelsbergh, M. 1994. Preprocessing and probing techniques for mixed integer programming. *ORSA Journal on Computing* 6(4):445–454.

van den Briel, M.; Vossen, T.; and Kambhampati, S. 2005. Reviving integer programming approaches for ai planning: A branch-and-cut framework. In *Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS-05)*, (to appear).

Wolfman, S., and Weld, D. 1999. The LPSAT engine and its applicationto resource planning. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-99)*, 310–317.