Finding Admissible Bounds for Over-subscription Planning Problems

J. Benton
Dept. of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287, USA
j.benton@asu.edu

Menkes van den Briel
Dept. of Industrial Engineering
Arizona State University
Tempe, AZ 85287, USA
menkes@asu.edu

Subbarao Kambhampati
Dept. of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287
rao@asu.edu

Abstract

When given a plan by a satisficing planner, it is usually not intuitive as to how close it is to the optimal solution. However, real world planning problems often require some metric on which to optimize, especially when goals are soft and resource constraints are involved, as is the case in over-subscribed planning problems. Unfortunately, little effort is given to find how close any plan may be to an optimal solution value. We set out to answer this shortcoming by providing a way to encode a relaxed version of over-subscribed planning problems in an integer program (IP) formulation. The solution to this formulation gives an admissible bound on the optimal solution value and can be further relaxed by dropping its integer constraints for better scalability.

Introduction

Though often it is not intuitive as to how close a given plan is to the optimal solution, the quality of solutions is of high importance for most real world planning problems. This is especially true when planning for problems in the presence of resource constraints and soft goals. In these cases, choice exists not only among the actions used to achieve goals, but also on the goals themselves. Most of the current state-of-the-art methods for solving these over-subscribed planning (OSP) problems return suboptimal solutions that attempt to minimize (or maximize) some quality objective. Such feasible plans may end up being useful and good enough for the task at hand. However, without information on the optimal plan quality value, we cannot know the "goodness" of a plan–that is, the distance between the optimal quality and the given plan. Though in general finding the optimal solution value is impractical (as hard as planning), there are other ways to assuage concerns of plan quality. In particular, it would be helpful to have a bound on the optimal solution value such that a user can make a more informed decision as to whether to spend additional computational resources to find a different plan.

A metric determining a plan's quality is typically given as part of the problem definition. Most widely studied over-subscription planning models involve a single objective that prioritizes goals. This is done in the partial satisfaction planning (PSP) model introduced in (Smith, 2004; van den Briel *et al.*, 2004) and version 3 of the planning domain description language (PDDL3), which was used in the 5^{th} International Planning Competition (IPC5) (Gerevini and Long, 2005).[1] Despite the use of these metrics there have been limited efforts to find domain-independent methods for finding a tight bound on the optimal solution in over-subscription planning. Instead, satisficing planners are compared against one another. And this is what was done in IPC5. While this was acceptable in that setting, using several planners for comparison can be infeasible in real world situations. Also, it fails to address fundamental questions on plan quality (e.g., whether all of the available planners giving poor solutions relative to optimal).

Consider the problem of scheduling travel to two cities, Metropolis and Capital City. Following the PSP model, reaching the cities is a soft goal and each is given a utility value. We want to maximize the difference between the utility given for reaching the cities and the travel cost, or a plan's *net benefit*. Capital City has a utility of 100 and Metropolis a utility of 500. Say we are given a plan for reaching both Metropolis and Capital City that costs 550, giving us a total *net benefit* of 50. Clearly we are achieving both goals and at first blush this appears to be a good plan. But the question is: how close to the optimal quality is this plan? It may turn out that visiting only Metropolis costs much less than visiting both locations, but there is no way for the user to know for sure. Using the current state-of-the-art planners a user can either (1) accept the given plan (2) find a new plan using a different planner or (3) in some planners, continue searching for a better plan using the current planner (e.g., this is possible in anytime planners like SPUDS (Do *et al.*, 2007) or HPlan-P (Baier *et al.*, 2007)). Since in this case the user has a maximization metric, it would be helpful to find some reasonably tight *upper bound* on the optimal value.

A well-known feature of admissible heuristic functions is that they can give a bound on potential plan quality.[2] While this observation is helpful, popular admissible heuristics for classical and temporal planning, such as the suite of heuristics used in HSP and its variants (Bonet and Geffner, 2001; Haslum, 2006), do not handle the balancing of goal and ac-

[1] Note that multi-objective, qualitative metrics have also been studied (Brafman and Chernyavsky, 2005).

[2] In classical planning problems, this is a lower bound as the metric usually involves minimizing either cost or the number of actions used.

tion choice as required with soft goals. Instead, a more sophisticated approach is needed for finding bounds in OSP problems. Indeed, though Do *et al.* (2007) discusses a way of combining the h_{max} heuristic with an integer program in an admissible heuristic called h^{GAI}_{max}, it does not detect the action dependencies for achieving each goal. In contrast, the heuristic proposed by Bonet and Geffner (2006) includes some action interactions but ignores static mutual exclusion between actions, which is important when there are action dependencies on goal achievement.

In this paper, we present a technique that utilizes the integer programming (IP) based heuristic introduced in Benton *et al.* (2007) and van den Briel *et al.* (2007) to calculate bounds on over-subscribed problems. The encoding relaxes the original problem by ignoring the ordering of actions while maintaining some causal information and the knowledge of negative effects that is encoded in SAS+ actions (Bäckström, 1992). In contrast to many similar approaches for encoding planning problems (c.f., van den Briel *et al.* (2005); Vossen *et al.* (1999)), this IP returns a value that does not use a predetermined number of steps in the resulting plan. Though integer programming is NP-complete to compute in general, we will see that the given formulation usually returns a reasonable bound on the optimal solution of problem instance. Additionally, we compare this with the encoding's linear program (LP) relaxation. The encoding is presented in the context of a generalized version of PSP, called PSP^{UD} and we expect that with some work it could be applied to PDDL3 temporal preferences and other over-subscription planning models. After introducing the encoding, we empirically show how well it scales by finding bounds for progressively more difficult problems.

Problem Formulation

In partial satisfaction planning (PSP), actions are given cost and goals are given utility values. Additionally, the requirement that all goals must be achieved is relaxed such that any subset of the goals may be achieved for a plan to be valid. By implication any sound plan is valid, including the empty plan. When a goal is achieved, its utility value is added to the quality of a plan and we want to maximize the difference between the utility received for achieving goals and the cost of the actions used in the plan, or the *net benefit*. For this work we use a generalization of PSP called partial satisfaction planning with utility dependencies, or PSP^{UD} (Do *et al.*, 2007). In this model, utility is assigned to sets of goals using k local utility functions, $f^u(G_k) \in \mathbb{R}$ on $G_k \subseteq G$, where any goal subset $G' \subseteq G$ has an evaluated utility value of $u(G') = \sum_{G_k \subseteq G'} f^u(G_k)$ as in the *general additive independence* model (Bacchus and Grove, 1995). Additionally, as in PSP, each action $a \in A$ has an associated cost $cost(a)$ such that $cost(a) \geq 0$.

Finding Bounds

Since the objective of PSP^{UD} is to *maximize* the value of *net benefit*, we find upper bounds on the optimal solution to problems. In problems where the objective is to *minimize* some quantity, we would find lower bounds.

We use the SAS+ representation of each planning problem and then encode the resulting values in an IP and add the PSP^{UD} properties (i.e., action cost and goal set utilities) in the IP formulation. Solving an IP is NP-complete in general, and so we can also find the LP relaxation of the IP encoding to determine upper bounds on the optimal solution of the original planning problem. Given a problem $\mathcal{P}$, the IP solution value, $IP_{\mathcal{P}}$, the LP solution value, $LP_{\mathcal{P}}$, and the optimal value $OPT_{\mathcal{P}}$, when maximizing on PSP^{UD} *net benefit* we will always have that $LP_{\mathcal{P}} \geq IP_{\mathcal{P}} \geq OPT_{\mathcal{P}}$. On minimization over-subscription planning problems the signs would be reversed.

Our IP encoding of planning problems does not involve the number of steps in the final plan. Because of this, we can ensure cost optimality. Other formulations involve specifying the number of steps for the encoding and are *bounded-length cost-optimal* in that they are only optimal given a specific plan length. However, this is done at the expense of finding a fully ordered plan. Instead, the IP that we will describe finds a set of actions.

The formulation requires as parameters: $cost(a)$, the cost of action a; $utility(v, f)$, the utility of achieving value f in variable v in the goal state; and $utility(k)$, the utility of achieving the goal utility dependency set G_k in the goal state. We also introduce the variables: $action(a) \in \mathbb{Z}^+$, the number of times action $a \in A$ is executed; *effect*$(a, v, e) \in \mathbb{Z}^+$, the number of times that effect e in state variable v is caused by action a; $prevail(a, v, f) \in \mathbb{Z}^+$, the number of times that the prevail condition f in state variable v is required by action a; $endvalue(v, f) \in \{0, 1\}$, a variable equal to 1 if value f in state variable v is achieved at the end of the solution plan (0 otherwise); $goaldep(k) \in \{0, 1\}$, a variable equal to 1 if goal dependency G_k is satisfied (0 otherwise).

The objective function maximizes the difference between the total utility given by the achieved goals and the total cost from the actions in the plan.

$$\text{MAX} \quad \sum_{v \in V, f \in D_v} utility(v, f) endvalue(v, f)$$
$$+ \sum_{k \in K} utility(k) goaldep(k) - \sum_{a \in A} cost(a) action(a)$$

- Action implication constraints for each $a \in A$ and $v \in V$. The SAS+ formalism allows the pre-conditions of an action to be undefined. We model this by using a separate effect variable for each possible pre-condition that the effect may have in the state variable. We must ensure that the number of times that an action is executed equals the number of effects and prevail conditions that the action imposes on each state variable.

$$action(a) = \sum_{\text{effects of } a \text{ in } v} \textit{effect}(a, v, e)$$
$$+ \sum_{\text{prevails of } a \text{ in } v} prevail(a, v, f)$$

- Effect implication constraints for each $v \in V$, $f \in D_v$. In order to execute an action effect its pre-condition must be satisfied. Hence, if we want to execute an effect that deletes some value multiple times, then we must ensure

Zenotravel	h^{GAI}_{max}	LP Value	IP Value	Optimal	h^{GAI}_{max} % Diff.	LP % Diff.	IP % Diff.
1	33267.89	19599.4	19599.4	19599.4	69.739%	0%	0%
2	37768.83	21047.13	21047.13	21047.13	79.449%	0%	0%
3	63417.52	56385.88	55304.18	54720.57	15.893%	3.04%	1.055%
4	63618.81	55214.68	53954.49	47887.8	32.850%	15.3%	11.244%
5	24182.08	25502.49	22713.18	22338.57	8.252%	14.163%	1.649%
Satellite	h^{GAI}_{max}	LP Value	IP Value	Optimal	h^{GAI}_{max} % Diff.	LP % Diff.	IP % Diff.
1	67042.48	67346.83	66979.51	66954.46	0.131%	0.586%	0.037%
3	109942.56	110142.67	109841.34	109820.97	0.102%	0.293%	0.009%
4	186576.17	186689.08	186348.1	186348.1	0.122%	0.183%	0%
6	166139.66	166263.28	165847.48	165798.03	0.206%	0.281%	0.03%
7	186363.20	186504.5	185949.23	185947.97	0.223%	0.300%	0.0007%
Rovers	h^{GAI}_{max}	LP Value	IP Value	Optimal	h^{GAI}_{max} % Diff.	LP % Diff.	IP % Diff.
1	40356.57	43005.695	37771.96	37771.96	6.843%	13.856%	0%
2	43862.03	43914.074	42460.01	42460.01	3.302%	3.311%	0%
3	42108	44085.62	36855.59	36589.75	15.081%	20.482%	0.726%
4	41864.34	43972.66	40189.09	40189.09	4.168%	8.604%	0%

Table 1: The upper bounds found on problems where the optimal solution is known.

that the value is added multiple times.

$$1\{\text{if } f \in s_0[v]\} + \sum_{\text{effects that add } f} \textit{effect}(a,v,e) = \sum_{\text{effects that delete } f} \textit{effect}(a,v,e) + endvlaue(v,f)$$

- Prevail implication constraints for each $a \in A$, $v \in V$, $f \in D_v$. In order to execute an action prevail condition it must be satisfied. Hence, if there is a prevail condition on some value, then that value must be added.

$$1\{\text{if } f \in s_0[v]\} + \sum_{\text{effects that add } f} \textit{effect}(a,v,e) \geq prevail(a,v,f)/M$$

- Goal dependency constraints for each goal dependency k. If all values of the goal dependency are achieved at the end of the solution plan, then the goal dependency is satisfied. Vice versa, if we want to satisfy the goal dependency, then we must achieve all its values at the end of the solution plan.[3]

$$goaldep(k) \geq \sum_{f \text{ in dependency } k} endvalue(v,f) - (|G_k| - 1)$$

$$goaldep(k) \leq endvalue(v,f) \quad \forall f \text{ in dependency } k$$

Results

To test our technique of finding bounds on the optimal solution values, several problem instances from domains in the 3^{rd} International Planning Competition (IPC3) were modified to include PSP^{UD} elements. Specifically, we made all goals soft, added costs to actions and added utility to goal sets in the STRIPS *zenotravel*, *rovers*, and *satellite* domains. While we did this on all 60 problem instances (20 problems $\times$ 3 domains), we had access to the optimal solution value in 14 of these (found using the planner BBOP-LP (Benton *et al.*, 2007)).

We implemented the encoding and solved both the IP and the LP relaxation of it using version 10 of the commercial CPLEX solver on a Pentium D 3.2Ghz processor. We compare this with the values found by h^{GAI}_{max} (Do *et al.*, 2007). For each problem Table shows the upper bound values found for *net benefit* at the initial state, as well as the optimal value. On these problems the upper bound value given by the IP solution is closer to the optimal value as compared with the other techniques. Figure 1 shows the time taken to solve for the upper bound using the LP relaxation. Figure 2 shows the same for the original IP formulation of the problem. As expected, the LP relaxation takes less time but it also provides worst bounds. Interestingly, the scale-up is quite good in both the *satellite* and *rovers* domains, but in *zenotravel* problems this is not the case.

To further examine the technique for finding bounds, we modeled the "MetricSimplePreferences" version of the Rovers domain from IPC5 in the IP encoding, the only "SimplePreferences" domain that includes explicit action cost (on the "navigate" action) and no disjunctive preferences.[4] In these problems the objective is a minimization (specified in a PDDL3 metric) and so we want to find *lower bounds* on quality. Table shows the results from this investigation. The bounds found using the IP formulation can be quite far from the optimal value. However, they are better than those given by h^{GAI}_{max}. They also can be found quite quickly (at most 1.4 seconds on our system for the displayed problems). Note that the LP relaxation returned a value of 0.0 in all of the problems. Since the current encoding can allow mutual satisfaction of constraints such that *flow in* and *flow out* of a value are equal it causes *disjoint flows* (e.g., flow that is not connected to the initial state). These disjoint flows will often provide optimal paths, affecting the solution values. In the modified IPC3 problems, disjoint flows occur less often.

[3] Note that both formulas are necessary since we may have negative values on goal utility dependencies (for problems with only positive values, the first formula would be redundant).

[4] We thank Patrik Haslum, who created this domain, for providing the optimal solution values to these problems.

Rovers-SP	h^{GAI}_{max}	IP Value	Optimal	h^{GAI}_{max} % Diff.	IP % Diff.
1	127.9	560.3	811.3	84.235%	30.938%
2	104.3	274.3	473.2	77.959%	42.033%
3	127.9	560.5	811.3	84.235%	30.938%
4	113.4	339.7	418.7	72.360%	18.868%
5	132.7	274.6	483.6	72.560%	43.218%
6	183.6	370.8	649.2	71.719%	42.884%
7	106.2	252.3	402.2	73.595%	37.27%

Table 2: The lower bounds found for some instances of the "MetricSimplePreferences" rovers domain from 5^{th} International Planning Competition.

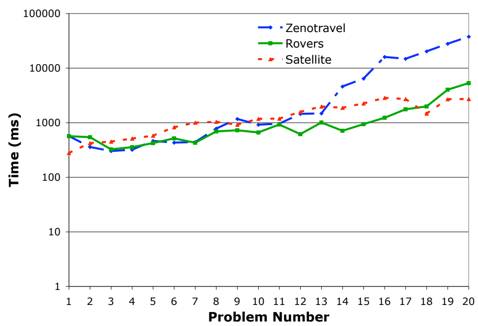

Figure 1: The time taken to solve for the LP relaxation upper bound values for problems in the *zenotravel*, *rovers*, and *satellite* domains.

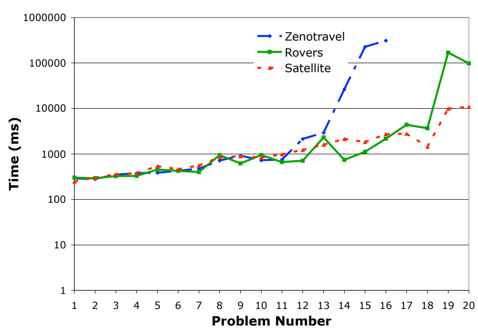

Figure 2: The time taken to solve for the IP upper bound values for problems in the *zenotravel*, *rovers*, and *satellite* domains.

Conclusion and Future Work

Many users of real-world planning systems want to know how good their plan is, and providing a means to determine this is of great benefit to them. We have presented our work on finding bounds on over-subscription planning problems, where plan quality is usually of particular interest. Interestingly, this encoding is successfully used in the planner BBOP-LP as a heuristic for solving PSP^{UD} problems (Benton *et al.*, 2007). Our investigation has uncovered some problems it may vastly over (or under) estimate potential plan quality. For our future work, we will put effort in determining ways of improving our encoding by including constraints that can help eliminate some of the issues we found.

References

F. Bacchus and A. Grove. Graphical model for preference and utility. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 3–10, 1995.

C. Bäckström. Equivalence and tractability results for SAS+ planning. In *International Conference on Principles on Knowledge Representation and Reasoning*, pages 126–137, Cambridge, MA, USA, October 1992.

J. Baier, F. Bacchus, and S. McIlraith. A heuristic search approach to planning with temporally extended preferences. In *Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 1808–1815, 2007.

J. Benton, M. van den Briel, and S. Kambhampati. A hybrid linear programming and relaxed plan heuristic for partial satisfaction planning problems. In *Proceedings of ICAPS*, 2007.

B. Bonet and H. Geffner. Planning as heuristic search. *Artificial Intelligence*, 129(1-2):5–33, 2001.

B. Bonet and H. Geffner. Heuristics for planning with penalties and rewards using compiled knowledge. In *KR*, 2006.

R. I. Brafman and Y. Chernyavsky. Planning with goal preferences and constraints. In *International Conference on Automated Planning and Scheduling (ICAPS)*, pages 182–191, 2005.

M.B. Do, J. Benton, M. van den Briel, and S. Kambhampati. Planning with goal utility dependencies. In *IJCAI-2007*, pages 1872–1878, 2007.

A. Gerevini and D. Long. Plan constraints and preferences in pddl3: The language of the fifth international planning competition. Technical report, University of Brescia, Italy, 2005.

P. Haslum. Improving heuristics through relaxed search - an analysis of tp4 and hsp*a in the 2004 planning competition. *Journal of Artificial Intelligence Research*, 25:233–267, 2006.

D. Smith. Choosing objectives in over-subscription planning. In *Proceedings of the 14th International Conference on Automated Planning and Scheduling (ICAPS-2004)*, pages 393–401, 2004.

M. van den Briel, R. Sanchez, M.B. Do, and S. Kambhampati. Effective approaches for partial satisfaction (oversubscription) planning. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI-2004)*, pages 562–569, 2004.

M. van den Briel, T. Vossen, and S. Kambhampati. Reviving integer programming approaches for AI planning. In *Proceedings ofICAPS-2005*, pages 310–319, 2005.

M. van den Briel, J. Benton, S. Kambhampati, and T. Vossen. An LP-based heuristic for optimal planning. In *Proceedings of the 13^{th} International Conference on Principles and Practice of Constraint Programming*, 2007.

T. Vossen, M. Ball, A. Lotem, and D.S. Nau. On the use of integer programming models in AI planning. In *Proceedings of IJCAI-1999*, pages 304–309, 1999.