Beyond Symbolic Goals: A Step Towards Utility-directed Planning *

Mike Williamson Steve Hanks
Department of Computer Science and Engineering, FR–35
University of Washington
Seattle, WA 98195
{mikew, hanks}@cs.washington.edu

Abstract

Classical AI planning adopts a very narrow notion of plan quality, namely that a plan is *good* just in case it achieves a specified *goal*. Despite the fact that planning is intractable in the worst case, goal-satisfying planning algorithms can effectively solve classes of problems by using the goal to focus the search for a solution (by using backward-chaining techniques), and by exploiting domain-specific heuristic knowledge to control search.

Our work extends the definition of plan quality to take into account partial satisfaction of the goal and the cost of resources used by the plan, while at the same time building an effective planning algorithm by exploiting classical planning techniques like backward chaining and knowledge-based search control rules.

This paper presents PYRRHUS, an extension to the UCPOP planning system [Barrett *et al.*, 1993] that finds optimal plans for a class of goal-directed utility models suggested by Haddawy and Hanks [Haddawy and Hanks, 1993]. Our empirical results suggest that optimal plans can be generated effectively by a planner using domain-specific heuristic knowledge, and furthermore that the planner can use the *same* knowledge as a goal-satisfying planner to solve corresponding optimization problems.

1 Introduction

The classical approach to AI planning has relied heavily on the notion of the *goal* – a symbolic description of the desired world state. Goals provide a valuable point of computational leverage, and have allowed the development of (heuristically) effective planning algorithms. But as the sole description of a planning problem they are in several regards inadequate. Goals include no notion of partial satisfaction; either a plan succeeds in achieving the desired world state or it does not. Goals also provide no way of describing the value of achieving the specified world state relative to the cost of achieving it. In short, goals provide insufficient information to measure the *quality* of a plan in any satisfying way.

Decision theory has been proposed as a solution to this problem. It provides a rich theoretical framework for the construction of utility models capturing precisely the notion of partial satisfaction and the cost-benefit tradeoff mentioned above. But the planning community has been slow to adopt utility models for the description of planning problems, since planning with utility models seems to require optimization, commonly thought to be less tractable and less amenable to heuristic control than classical goal satisfaction.

Our claim is that optimal planning can be made heuristically tractable for utility models that provide significantly more expressive power than simple goal formulas. In this paper we adopt a particular utility model that allows description of *soft deadline* goals (whose value is a function of satisfaction time) and plan cost as a function of resource consumption. As in classical planning, we achieve heuristic tractability by exploiting the structure of the utility model and by applying domain-specific search control knowledge to guide the planning process. We have implemented PYRRHUS, a least-commitment planning algorithm that finds optimal plans for this utility model in time comparable to that of a classical planning algorithm, given equivalent heuristic search control knowledge.[1]

The remainder of this paper describes the utility model we are investigating, the PYRRHUS planning algorithm, and our preliminary experimental results.

2 The PSTC utility model

Haddawy and Hanks [Haddawy and Hanks, 1993; Haddawy and Hanks, 1992] discuss in detail the problem of constructing utility models for *goal-directed* agents, and suggest several alternatives of varying expressive power. We have adopted a variant of one of their proposals and dubbed it the *PSTC* (Partial Satisfaction of the Temporal Component) utility model. This section describes that model in detail.

*Thanks to Denise Draper, Nick Kushmerick, Neal Lesh, and Dan Weld for useful and stimulating feedback. This work was supported in part by NSF grant IRI–9008670 and by an NSF Graduate Research Fellowship.

[1] Note that we are not employing decision theory to represent or reason about *uncertainty*. We are interested here only in the use of utility theory to characterize differences in the quality of alternative plans.

Haddawy and Hanks characterize deadline goals as having two components: an atemporal component (describing *what* must done) and a temporal component (describing by *when* it must be done). In our model, we are not allowing partial satisfaction of the atemporal component, so it can be represented by a symbolic description of the desired world state, equivalent to a goal from classical planning. Indeed, we will sometimes refer to the atemporal component of the deadline goal as the "goal formula."

The temporal component (i.e. deadline) does allow partial satisfaction. It is represented by a *temporal decay function* conveying the deadline and the relative value of satisfying the goal thereafter. This is a function from the earliest time that the goal is attained onto the unit interval, subject to the constraints that before the deadline it is uniformly 1.0, and after the deadline it is monotonically non-increasing.

The other major component of our utility model is a *residual utility* term, summarizing the effect of a plan on non-goal attributes. We describe residual utility in terms of *assets*, an application-identified subset of the metric resources in the domain. Similar in character to temporal decay, residual utility is represented by a function from the net amount of each asset consumed (or produced) to the unit interval. The function must satisfy the constraint that it is monotonically non-increasing with respect to any given asset (i.e. all other things being equal, consuming an asset does not increase utility).

The final component of the utility model is a *goal value coefficient* that determines the relative value of achieving the goal with respect to the consumption of resources.

A PSTC utility model, then, consists of four components: the goal formula, G; the temporal decay function, $T(t)$; the residual utility function, $R(a_1, a_2, \ldots)$; the goal value coefficient, k_G.

An initial state description and a utility model together define a planning scenario. We define the utility of a plan to be the utility of the outcome of the execution of that plan starting from the initial state. Since we do not model uncertainty or exogenous events, there is a single possible outcome for any plan, so we can calculate the actual rather than the expected utility. The utility of such an outcome, o, is defined to be:

$$U(o) = k_G \cdot S_G(o) \cdot T(E_G(o)) + R(c_1(o), c_2(o), \ldots)$$

where $S_G(o)$ is 1.0 if G is satisfied in o, and 0.0 otherwise; $E_G(o)$ is the earliest time that G is satisfied in o; $c_i(o)$ is the net amount asset a_i changed in o.

3 The PYRRHUS planning system

Given a planning problem composed of an initial state description and a PSTC utility model, we would like an algorithm that can find an optimal plan. Of course, given that classical planning is intractable even under limiting assumptions [Bylander, 1991], and considering that PSTC optimal planning is at least as hard as classical planning,[2] we cannot expect any algorithm to perform well on all problems. The best we can hope for is an algorithm that is *heuristically tractable*, that is, one that can perform well on a class of problems in a particular domain, given an adequate body of domain-specific control knowledge. In this section we present the PYRRHUS planning algorithm, and describe how it achieves heuristic tractability by 1) exploiting the structure of the utility model, and 2) employing domain-specific search control knowledge.

[2]This is true in the formal sense — any classical planning problem can be expressed using a PSTC utility model.

3.1 Action Representation

The underlying planning mechanism of PYRRHUS is based on the UCPOP planning system [Penberthy and Weld, 1992; Barrett *et al.*, 1993], which is itself derived from the SNLP causal-link planner [McAllester and Rosenblitt, 1991]. PYRRHUS uses the UCPOP 2.0 action representation language for variablized operator schemata, including universal quantification and conditional effects. We made extensions to provide (rudimentary) support for time and metric resources.

Resources Planning with metric (consumable) resources is a challenging problem in its own right, and has been the focus of significant research attention [Penberthy, 1993; Wilkins, 1988]. Although our main interest is in planning with utility models rather than planning with resources *per se*, the two are related; the quality of a plan usually depends on the resources it consumes.

Metric resources are represented in PYRRHUS by a set of symbolic *resource designators*. The initial state contains *initial resource descriptors*, for example, `(:resource fuel-level 10)` indicating that there are initially 10 units of fuel in the fuel tank. Operators can have *resource preconditions* of the form `(<= 15 fuel-level 20)` requiring the fuel level at the outset of the action to be between 15 and 20 units, and can have *resource effects* of the form `(fuel-level -15)` indicating that 15 units of fuel are consumed by the operation. In general, actions can both produce and consume resources, subject to some restrictions on the production of assets, described in detail below.

Note that this model of resources is somewhat restrictive: its primary limitation is that the amounts specified in the preconditions and effects must be numeric constants, and cannot, therefore, depend on the bindings of variable parameters in the operator schemata. However, we believe that there would be no conceptual impediment to incorporating any of the more advanced resource reasoning techniques from classical planning [Penberthy, 1993; Wilkins, 1988; Currie and Tate, 1991].

Time Time could be treated like any other metric resource, with an action consuming an amount corresponding to its duration. But time has some distinguishing characteristics, namely 1) all actions consume some amount of time, 2) no actions *produce* time, and 3) in our model, no actions have temporal preconditions (i.e. any action can be executed at any point in time, subject to step-ordering constraints). These three characteristics collectively allow for specialized temporal reasoning that is far more efficient than our generalized metric resource reasoning. For this reason, time is distinguished

from other resources, and each operator schema has a specified *duration*. As with other metric resources this duration must be a numeric constant.

3.2 Optimal PSTC planning

The PYRRHUS algorithm is a synthesis of branch-and-bound optimization with a least-commitment, plan-space planner. Much like a classical (goal-satisfying) causal-link planner, PYRRHUS begins by initializing a plan queue to contain an initial partial plan, which is built from the initial state and the goal formula from the utility model. PYRRHUS iteratively removes a partial plan from the queue, generates its refinements, and returns them to the queue. A plan is refined by choosing one of its flaws, and generating all possible fixes for that flaw. There are two types of flaws. An *open condition* represents an unsatisfied precondition of a plan step, and is fixed by adding a *causal link* from an effect of either a new or an existing plan step. A *threatened link* is a causal link whose condition may conflict with an effect of some *clobbering* step. Threatened-link flaws are fixed by adding plan step ordering constraints or variable binding constraints to prevent the conflict from occurring. See [Weld, 1994] for a detailed description of least-commitment planning with causal links.

Establishing dominance The difference between PYRRHUS and a goal-satisfying planner is that while the latter can stop as soon as a complete (flawless) plan is found, the former cannot. But PYRRHUS need not explicitly consider *every* possible complete plan — rather, it need only establish that the plan it returns is not *dominated* by any others. The key to establishing dominance is the determination of upper bounds on the utility of partial plans. The definition of the PSTC utility model implies that increasing the amount of time taken or the amount of assets consumed by a plan can only decrease its utility. So, given a partial plan, if we can determine a lower bound on the time and asset consumption of all possible refinements of that plan, we can consequently determine an upper bound on possible utility.

To obtain a lower bound on time and asset consumption, we require that the domain adhere to a condition of strict *asset-position monotonicity*, that is, that no operator is allowed to increase any asset without decreasing some other asset so as to not increase residual utility. For example, if residual utility depends on two resources, `fuel-level` and `money`, then a refueling operator that increases `fuel-level` would have to decrease `money` by some amount so as not to increase residual utility. Asset-position monotonicity is a strong way of saying "You can't get something for nothing."[3]

In general, it is not possible to find a meaningful *lower* bound on the utility of a partial plan, since there will be some refinements that make a plan arbitrarily bad. Lower bounds can only be determined for complete plans, for which an exact utility can be computed.

PYRRHUS establishes an initial greatest lower bound (GLB) on the utility of any eventual solution by calculating the utility of the *status quo* directly from the utility model and the initial state. The utility of the *status quo* corresponds to never achieving the goal, but expending no resources either. In the course of planning, each time a complete plan is found its (exact) utility is determined and compared to the GLB. When a plan better than the GLB is found, the GLB is updated, and that plan is saved as the best so far. Whenever the GLB is increased, all plans in the queue whose upper bound on utility is less than the new GLB are discarded. Planning continues in this manner until the plan queue is empty, ensuring that the plan that established the greatest lower bound on utility is optimal.

3.3 Efficiency of the planner

The essential question, of course, is whether this optimal planning algorithm — which is clearly exponential in the worst case — can be rendered tractable on useful classes of problems through the application of domain-specific heuristics. On the one hand, the nature of the optimization makes it seem as if more partial plans will need to be examined. On the other hand, information from the utility model allows pruning plans from the search space that a classical planner might need to consider. These two tendencies are in opposition, and the nature of the domain and (more importantly) the utility model determines which will dominate.

As in goal-satisfying plan-space planning, there are two points in the algorithm at which domain-specific knowledge can be applied. The first is in the choice of a plan to refine, which determines the path of the search through the space of partial plans. The second is in the choice of which flaw in that plan to fix, which determines the children of that plan and thus the branching factor of the search space. In classical planning, the value of making these choices correctly is clear: the planner is led directly to a solution and so can halt. It is less obvious that this kind of heuristic knowledge is valuable to an optimizing planner, which cannot necessarily halt when it finds a complete plan. We found, though, that heuristic search-control knowledge *can* be useful for optimization with goal-directed utility models. PYRRHUS uses the same representation for heuristic knowledge as a classical planner, and in our experience many of the rules helpful to classical planning aid optimization as well. By finding a complete plan more quickly, PYRRHUS can establish a higher greatest lower bound on utility, allowing it to prune more partial plans before they are expanded.

PYRRHUS inherits its heuristic search control mechanism from UCPOP [Barrett *et al.*, 1993]. It is a rule-based production system, inspired by that of PRODIGY [Minton *et al.*, 1989], and allows specification of declarative domain-dependent rules to control the choice of which plans to expand and which flaws to fix. The antecedents of the rules match characteristics of the plans and/or flaws under consideration, and the consequents assert preferences among those candidates. Experience in classical planning has been that relatively small bodies of this kind of knowledge can often allow a planner to quickly solve broad classes of problems. We have observed similar results in optimal planning; most (though

[3] In fact, we think this is *too* strong a way of saying it, and will address the need for this restriction in future research.

not all) of the rules that benefit a classical planner for our domain are beneficial to PYRRHUS also.

4 Experimental Results

We are evaluating PYRRHUS using a number of experiments in a simplified transportation planning domain inspired by the Truckworld planning testbed [Hanks *et al.*, 1993; Nguyen *et al.*, 1993]. The domain consists of a graph of locations connected by roads of various lengths. Some of the roads are toll roads, requiring payment in cash before use. A robot truck is given tasks of delivering objects between various locations in the world. The planner must determine a route for the truck, allowing it to pick up and deliver the specified objects. The planner must also decide whether the truck should drive slow or fast over each individual road segment. Driving fast allows quicker delivery but consumes more fuel. The planner must reason about the fuel usage of the truck, and insert refueling actions into the plan, taking into account the location of fuel drums in the world. For any given problem there are a number of *feasible* solutions which achieve the goal with no redundant steps. These differ in the route the truck takes, the speed it drives, and where it refuels.

For comparative purposes, PYRRHUS can be made to function as a classical goal-satisfying planner, in which it is just given a symbolic goal instead of a utility model. In this mode, PYRRHUS operates like UCPOP [Barrett *et al.*, 1993], except for the extensions to handle metric preconditions and effects. Our basic methodology was to compare optimal planning to classical planning in our domain.

4.1 Experiment One

We constructed a suite of 13 classical (atemporal) planning problems of varying difficulty for this domain. All problems started with the same initial state, but harder ones required delivery of more objects over greater distances. The easiest problem could be solved with a plan of four steps, while the most difficult one required fourteen steps. Only the easiest four problems could be solved by the classical planner without the use of domain specific search control.[4] A small body of four search control rules, however, allowed all the problems to be solved quickly.

We constructed two different PSTC utility models for each classical planning problem. In each case, the residual utility function was the same, decreasing linearly in the amount of fuel and money used[5]. The temporal decay function always decayed linearly from 1.0 to 0.0 over 100 time units after the deadline, but the deadline varied in each case. The class of problems we call *PSTC Late* set the deadline for each problem late enough that all feasible plans could meet the deadline. The *PSTC Early* problems, in contrast, set the deadline early enough that all feasible solutions would miss it to some degree. The goal value coefficient was set so that achieving the goal by the deadline was equal in utility to $50.00 worth of fuel and money.

Figure 1 shows the number of partial plans explored[6] by PYRRHUS on the classical, *PSTC Early*, and *PSTC Late* variations of each of the thirteen problems. The notable feature of this graph is that the *PSTC Late* problems are uniformly much easier than their classical counterparts, while the *PSTC Early* problems are sometimes more difficult. This difference in difficulty arises because all feasible solutions for the *PSTC Late* problems occur before the deadline. Hence the utility of these plans depends only on the amount of fuel consumed, so the plan to drive slowly over the shortest route clearly dominates all the others. The information available in the utility model allows pruning of alternatives that the classical planner would consider.

In the *PSTC Early* case, however, all the feasible plans achieve the goal after the deadline. This gives rise to a utility tradeoff: driving faster allows earlier satisfaction of the goal, but at a cost in fuel, and hence a corresponding decline in residual utility. (The optimal solution depends on the ratio of residual and goal coefficients.) The lack of a clear optimum requires the planner to expand partial plans in greater detail in order to establish dominance.

4.2 Experiment Two

In an effort to explore more thoroughly the relationship between the nature of the utility model and the difficulty of optimal planning, we focused on a single problem (the hardest classical planning problem) and varied the utility model by changing the character of the temporal decay function. There are seven feasible solutions to this problem varying from 310 to 350 time units in duration, and from 14 down to 10 gallons of fuel consumed. We describe the temporal decay function using pairs of the form *(d k)* to indicate a deadline of d, with goal utility decaying linearly to 0.0 over a period of k time units after the deadline. The case of $k = 0$ indicates a hard deadline, after which there is no utility at all in satisfying the goal.

Figure 2 shows the results of this experiment. The two problems marked with a "*" were ones for which no plan could be established as optimal within the search limit of 20,000 partial plans. In some cases, such as those with a hard deadline that could not be met by any feasible plan, the optimal plan was correctly established to be the status quo.

Inspection of Figure 2 shows that the difficulty of optimal planning strongly depends on the character of the utility model. In many cases, the information provided by the utility model was very beneficial, allowing an optimal plan to be found more quickly than the classical planner could find *any* plan. The two problems that could not be solved shared a common characteristic: in both cases there were a large number of partial plans whose utility could not be shown to be dominated until they had been thoroughly refined. The problem *(300 0)* was particularly pernicious. This is a hard deadline

[4] Given a search limit of 20,000 partial plans.

[5] Based on a resource equivalence rate of $1.00 per gallon of fuel.

[6] The CPU time per partial plan explored is comparable for both classical and optimizing planning.

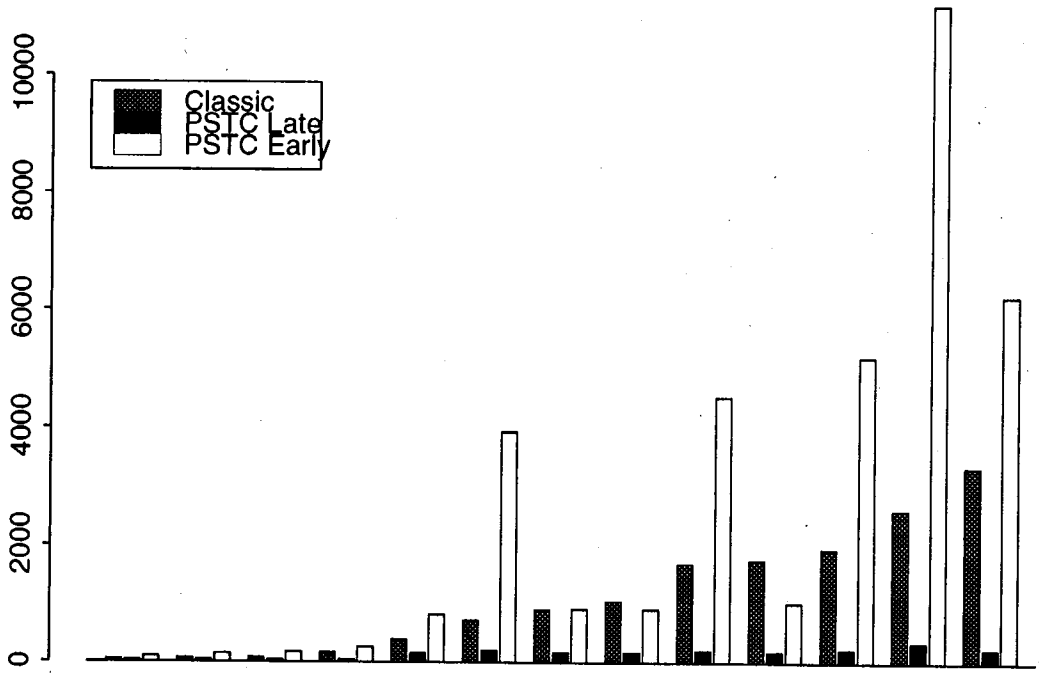

Figure 1: Number of partial plans explored for classical planning and two different utility models on a variety of problems.

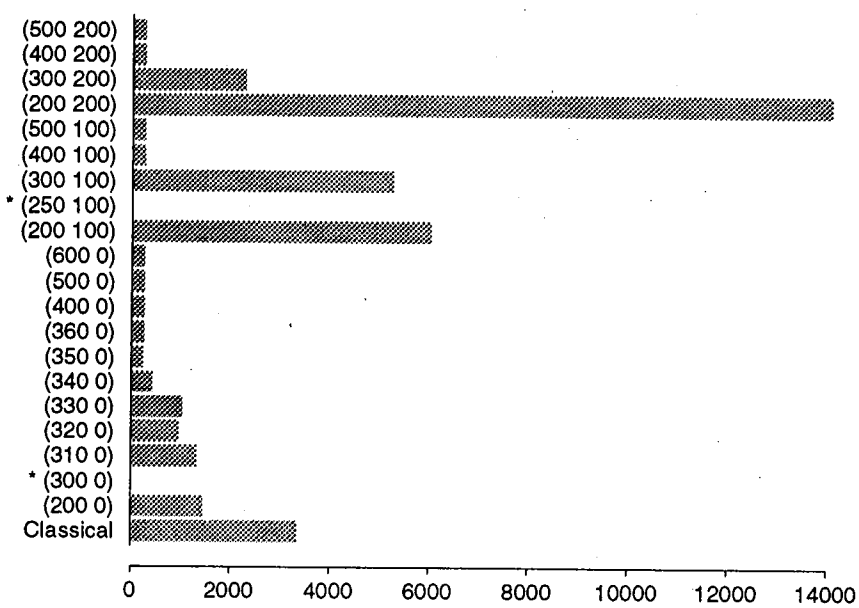

Figure 2: Number of partial plans explored for several different utility models on a given problem.

just short of the time required by the fastest plan. In essence, many partial plans looked quite good — only when the final action was added (pushing them past the deadline) did their utility drop abruptly. Increasing the search limit showed that 56305 plans would have to be explored to solve this problem, making it about one order of magnitude harder than classical planning.

5 Related and future work

Feldman and Sproull [Feldman and Sproull, 1977] were among the first to suggest the use of decision theory in AI. They suggested the essential idea of using bounds on utility to guide the planning process, but they focused on the representation of uncertainty, leaving their utility model relatively undeveloped. Wellman [Wellman, 1990] advanced the notion of proving dominance between classes of plans by constructing qualitative decision models. Neither of the above addressed the role of heuristic knowledge, or undertook any empirical investigation of the tractability of decision-theoretic planning. Etzioni [Etzioni, 1991] advocated the use of decision theory for making control decisions, but was not doing planning in that methods for achieving each goal were specified in advance. Similarly, optimization techniques from the field of operations research [Bradley *et al.*, 1977] generally require prespecified alternatives, and are not amenable to knowledge-based problem-solving techniques such as symbolic search control.

Future directions We still need to explore the applicability of the PSTC utility model to different domains, and in particular, to assess the performance of PYRRHUS when applied to realistic problems. We would also like to consider more expressive utility models, and the adaptation of other classical planning techniques (such as HTN planning) to utility optimization. Finally, we would like to explore approximate optimization techniques for finding *good* plans when finding the *best* plan is too expensive.

6 Conclusions

We have presented a novel planning algorithm that finds optimal plans for a utility model more expressive than simple goal formulas. The algorithm achieves heuristic tractability by taking advantage of the goal component of the PSTC utility model and by employing the kind of search control knowledge used by classical planners. Our initial experimentation supports our claim that optimal planning *need not* be more difficult than classical goal-satisfying planning, but instead that the information provided through the utility model can sometimes make planning easier. We are in the process of developing a formal characterization of the relationship between the structure of the utility function and the difficulty of optimal planning.

References

[Barrett *et al.*, 1993] A. Barrett, K. Golden, J.S. Penberthy, and D. Weld. UCPOP user's manual, (version 2.0). Technical Report 93-09-06, University of Washington, Department of Computer Science and Engineering, September 1993. Available via anonymous FTP from `~ftp/pub/ai/` at `cs.washington.edu`.

[Bradley *et al.*, 1977] S. P. Bradley, A. C. Hax, and T. L. Magnanti. *Applied Mathematical Programming.* Addison-Wesley, 1977.

[Bylander, 1991] T. Bylander. Complexity results for planning. In *Proceedings of IJCAI-91*, pages 274–279, 1991.

[Currie and Tate, 1991] K. Currie and A. Tate. O-plan: the open planning architecture. *Artificial Intelligence*, 52(1):49–86, November 1991.

[Etzioni, 1991] Oren Etzioni. Embedding decision-analytic control in a learning architecture. *Artificial Intelligence*, 49(1–3):129–160, 1991.

[Feldman and Sproull, 1977] J. Feldman and R. Sproull. Decision theory and artificial intelligence II: The hungry monkey. *Cognitive Science*, 1:158–192, 1977.

[Haddawy and Hanks, 1992] P. Haddawy and S. Hanks. Representations for Decision-Theoretic Planning: Utility Functions for Dealine Goals. In *Proc. 3rd Int. Conf. on Principles of Knowledge Representation and Reasoning*, 1992.

[Haddawy and Hanks, 1993] Peter Haddawy and Steve Hanks. Utility Models for Goal-Directed Decision-Theoretic Planners. Technical Report 93–06–04, Univ. of Washington, Dept. of Computer Science and Engineering, September 1993. Available via anonymous FTP from `~ftp/pub/ai/` at `cs.washington.edu`.

[Hanks *et al.*, 1993] Steve Hanks, Dat Nguyen, and Chris Thomas. The truckworld planning testbed, August 1993. Submitted to *IEEE Expert.*

[McAllester and Rosenblitt, 1991] D. McAllester and D. Rosenblitt. Systematic nonlinear planning. In *Proc. 9th Nat. Conf. on Artificial Intelligence*, pages 634–639, July 1991. internet file at ftp.ai.mit.edu: /pub/users/dam/aaai91c.ps.

[Minton *et al.*, 1989] S. Minton, C. Knoblock, D. Koukka, Y. Gil, R. Joseph, and J. Carbonell. PRODIGY 2.0: The Manual and Tutorial. CMU-CS-89-146, Carnegie-Mellon University, May 1989.

[Nguyen *et al.*, 1993] Dat Nguyen, Steve Hanks, and Chris Thomas. The truckworld manual. Technical Report 93–09–08, Univ. of Washington, Dept. of Computer Science and Engineering, September 1993.

[Penberthy and Weld, 1992] J.S. Penberthy and D. Weld. UCPOP: A sound, complete, partial order planner for ADL. In *Proc. 3rd Int. Conf. on Principles of Knowledge Representation and Reasoning*, pages 103–114, October 1992. Available via anonymous FTP from `~ftp/pub/ai/` at `cs.washington.edu`.

[Penberthy, 1993] J.S. Penberthy. *Planning with Continuous Change.* PhD thesis, University of Washington, 1993. Available as UW CSE Tech Report 93-12-01.

[Weld, 1994] D. Weld. An introduction to least-commitment planning. *AI Magazine*, To appear Summer or Fall 1994. Available via anonymous FTP from `~ftp/pub/ai/` at `cs.washington.edu`.

[Wellman, 1990] Michael P. Wellman. *Formulation of Tradeoffs in Planning Under Uncertainty.* Pitman Publishing, London, 1990.

[Wilkins, 1988] D. E. Wilkins. *Practical Planning.* Morgan Kaufmann, San Mateo, CA, 1988.