Some Issues in Practical Planning with Implications for Learning

Paul E. Lehner

Department of Systems Engineering
George Mason University
Fairfax, Virginia 22030
& MITRE Corporation
plehner@gmu.edu

Christopher Elsaesser

Advanced Information Technology Center
MITRE Corporation
7525 Colshire Drive
McLean, Virginia 22102
chris@azrael.mitre.org

Abstract

Several areas are identifed where machine learning procedures could be employed to substantially improve the quality of practical planners. All of these areas relate to the use of machine learning to define abstraction levels and to control search.

Introduction

We have been involved in a variety of efforts that involve the application of knowledge-based planning technology to realistic domains. Based on this experience, we have identified several issues in realistic planning that offer opportunities for the integration of planning and learning, but are often overlooked by the theoretical research community.

Experience

Below is a list of planning systems we have or currently are developing. This list only includes systems that satisfy one of the following criteria.

1. The system is or is intended for operational system.
2. The system has been applied to a user-selected operational problem
3. The system was designed to be evaluated a human competitor.

Battle planning. Battle planns specify the simultaneous coordinated movement of multiple ground units in a hostile environment. In this domain we have implemented systems that reflect two different approaches. The first uses a generative planning where the planner is recursively called to hypothesize actions for an adversary that will deny a precondition in the friendly plan (Elsaesser and Macmillan, 1991). The second is based on game playing techniques, where search procedures are applied to a knowledge-generated search space (Lehner, 1990).

Real-time planning and control of submarine combat actions. Here a reactive planner controls simulated adversaries in combat training (Anton, et. al. 1994).

Real time planning and control of self defense actions. Self defense assets are allocated in a few seconds against multiple incoming missiles. This planner is a transformational planner that begins with an "idealistic" unconstrained allocation plan and heuristically searches through a space of plan modifications.

Sensor planning. The actions of multiple mobile sensors are planned to search a large search area for agents that are trying to avoid detection (Musman, et.al., 1994). First, possible maneuver plans for the elusive agents are hypothesized. Relevant features of these plans, combined with the results of terrain and sensor analyses, are summarized as a Bayesian network for evaluating sensor plans. Depth first iterative broadening is used to search through the space of possible sensor plans, where heuristic search is guided by evaluating the Bayesian network.

Route planning. Simulated annealing techniques are used to control search through a space of heuristically-generated route modifications.

Strategic action planning. This system is designed to analyze the impact of sequences of strategic actions (economic embargoes, military support, trade relations.) designed to affect the political situation in a selected region of the world.

Explosive ordnance disposal. Helps a person plan a sequence of actions to disarm an explosive.

Opportunities to Integrate Practical Planning and Learning.

Based on our experience with the practical application of knowledge-based planning technology, we have identified several learning issues that are important to the

development of practical planning systems, but seem to be largely overlooked in the theoretical community. We will use examples from the maneuver planning domain to illustrate some of our points.

Plans are irrelevant - Planning is essential.

With the exception of explosive ordnance disposal, all of the above systems involved planning in a dynamic environment. Rarely in such environments does one expect the original plan to succeed. Indeed, it is common to hear comments of the form "On the first day of the war, you throw out all the plans." Despite this, interest in generating plans is ubiquitous. There are two related benefits to planning other than the plan generated: the plan gets an agent started in a direction that will eventually lead to success, the planner learns something about the planning problem that is used later during replanning. Each of these are discussed below.

Plan success vs. plan leading to success. In dynamic environments, as the size and complexity of plans increase, the probability that those plans will be executed successfully quickly approaches zero. However, a good plan may have a high probability of leading to a success, because it can be adapted as a situation unfolds. For hierarchical plans, this often implies that the probability that a plan will lead to success is determined principally by the probability that the higher level elements of the plan can be accomplished even when lower level actions fail. This implies that for a planner to estimate the probability of a plan leading to success, the planner must estimate its own ability to find effective plan repairs in unexpected circumstances. We have proposed two approaches to estimating the probability of plan repair (Lehner and Elsaesser 1993; Lehner, et.al., 1994). Both approaches require, for each operator in the knowledge base, a model that estimates the probability that a successful subplan for that operator will be found. Engineering such probability models is a difficult task, since each model defines an expectation over how the planner will behave. A more practical approach would be to implement a learning procedure that develops these probability models from experience gained from applying the planner to real or simulated tasks.

Problem-specific learning. Each planning problem has unique characteristics. In generating a plan, the planner has an opportunity to gather information about many characteristics of the current planning problem that may not have been apparent when the problem was specified (problem areas, solution density, resource criticality, etc.). These discovered characteristics can be used to guide replanning or plan repair. For instance, if a proposed manuever plan nearly exhausts a resource (e.g., artillery assets), then a reasonable replanning strategy is one that was conservative in its allocation of that resource (e.g., use Close Air Support rather than artillery). Learning procedures that can extract from a specific planning instance information that generalizes to replanning would have two uses: tailoring the replanning strategy and revising the estimate that a planner can successfully revise a plan as the situation unfolds.

Learning Abstraction Levels

Defining appropriate abstraction levels is key to the success and performance of many planning systems. There are some interesting opportunities to apply machine learning in defining abstraction levels for realistic planners.

Defining the primitive level. The primitive abstraction level includes the lowest level elements from which a problem description and plan are built. The problem of finding the most appropriate level of abstraction for the primitives has been largely ignored in the knowledge-based planning literature. Many seem to view this problem as a tradeoff between accuracy and complexity. In this view, adding detail to the primitive abstraction level increases the accuracy of the planner's projection, but it also increases the time required to generate plans. This view is only partially correct. It is easy to construct counterexamples where a detailed model of a domain will lead to less accurate projections than a superficial model, even if the detailed model is a valid decomposition of the superficial model. Consider, for instance, a battle outcome estimator that used force ratios to predict the "winner" of a conflict. Suppose the outcome calculator is 90% accurate. Consider now the choice of planning at the division level, or planning at the brigade level. At the division level a single force ratio calculation is required with an accuracy of 90%. At the brigade level, the division area is decomposed into three seperate conflicts. Predicative accuracy of each conflict is 90%, but it is known with certainty that whichever side wins two of the three brigade level conflicts will win the division level conflict. This implies the following calculations.

P("W"|W)
= P(www)·P("W"|www)
+ P(wwl or wlw or lww)·P("W"|wwl or wlw or lww)
= .25·.972 + .75·.828 = .864,
where

W = win at division level,

"W" = win at division level is predicted,

www, wwl,... = set of wins and losses at the brigade level, and

prior probability of a win at any level is .5.

Predictive accuracy decreased even though it is based on a perfect model of the relationship between abstraction levels! The predictive accuracy decreases quickly as the division level conflict is futher decomposed. This phenomena occurs because adding detail to a model increases the number of parameters that must be assessed, which increases the number of possible sources of error.

Except for features that the planner must include because of the nature of the planning task, additional features should only be included if the ability to project the value of that feature offsets the errors that might be introduced if that feature is inappropriately predicted. Note that this sometimes implies (paradoxically, but correctly) that features that are known to be key to determining an outcome should not be modeled, if the ability to predict that feature is poor.

Although predictive accuracy of individual features can be knowledge engineered, it is clear that the estimate of predictive accuracy should be monitored and updated over time. Learning could be used to modify the elements that are included in the primitive abstraction level of the planners problem representation.

Hierarchical abstraction levels. Many practical planners plan hierarchically. Many, for instance, develop an initial plan by considering only a subset of the problem variables and then add details to the plan by iteratively expanding the set of variables they consider. Typically, the set of variables in each abstraction level is knowledge engineered. However, this issue is very amenable to machine learning. Specifically, the set of variables in an abstraction level is appropriate if (1) plans generated by considering the higher abstraction levels are not invalidated (or substantially devalued) when considering the current abstraction level, and (2) the plans that result from the current abstraction level remain valid when considering the lower abstraction levels. It should be straightforward to develop a learning procedure that learns appropriate abstraction levels by applying a hierarchical planner at different abstraction levels. This type of learning would be pariticularly appropriate if a domain simulation is available. Alternative abstraction hierarchies could be tested in the domain simulation to determine abstraction levels that required the least amount of backtracking. Unless the domain simulation is a very poor representation of reality, it is likely that the abstraction hierarchy be appropriate for realistic problems. Indeed, the robustness of a proposed abstraction hierarchy could be included as part of the simulation study.

Although there has been some work in this area (Knoblock, 1990), it was limited to planners that do not consider context-dependent effects. This is innappropriate for most realistic domains.

Problem-specific abstraction levels. The procedure for learning abstraction levels can also be made problem specific. That is, if a planner using one set of abstraction levels is forced to backtrack because of a problem at a lower abstraction level, then the learning procedure can be invoked to identify the variables that were the source of the problem. These variables can then be moved to a higher abstraction level. This is an instance of problem specific learning that was described above.

Actions with multiple-purposes

Finally, we have discovered that many domains require actions that simultaneously support achievement multiple goals. While it is relatively easy to engineer actions that achieve a single purpose, multi-purpose actions often reflect a situation-specific merging of multiple single-purpose actions. Such multi-purpose actions are hard to engineer *a priori*. This suggests that a practical planner should have some ability to syntactically combine actions. Unfortunately, the number of potential combinations is often rather large and procedures need to be developed to strongly bias the planner toward effective combination procedures. These biases could be learned.

Summary: Learning Planning Strategies

We have identified several areas where machine learning procedures could be employed to improve the quality and performance of practical planners. Most of the above suggestions involve the application of machine learning techniques to redirect a planner's search procedure. Consequently, these suggestions are instances of the notion of learning planning biases as outlined in Rosenbloom, et. al. (1993).

References

Anton, P., McAdow, K., Smith, D., and Lehner, P. (1994) "Automatic Interactive Targets for Submarine Training", in *Proceedings of the 1994 TECOM Artificial Intelligence Technical Symposium*, Aberdeen Proving Ground, MD, September 1994.

Elsaesser, C. and Macmillan, T.R., *Representation and Algorithms for Multiagent Adversarial Planning*, Technical Report MTR-91W000207, MITRE Corporation, December 1991.

Knoblock, C. (1990) "Learning Abstractin Hierarchies for Problem Solving," in *AAAI-90*, 923-928.

Lehner, P. (1990) "Automated adversarial planning search procedures with provable properties," in Advanced Technology for Command and contrl systems Engineering, S. Andriole (ed.), AFCEA Intenational Press: Fairfax, VA.

Lehner, P. and Elsaeser, C. (1993) "Probabilistic reasoning in a classical hierarchical planner," Proceedings of the Second International Symposium on Uncertainty Modeling and Analysis, April 1993.

Lehner, P., Elsaeser, C. and Musman, S. (1994) "Constructing belief networks to evaluate plans," in *Proceedings of the 1994 Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, July 1994.

Musman, S., Lehner, P., Elsaeser, C. (1994) "Searching for Elusive Agents," in working notes of the *AAAI Workshop on Spatial and Temporal Reasoning*, July 1994.

Rosenbloom, P., Soowon, L. and Unruh, A. (1993) Bias in Planning and Explanation-based Learning. In S. Minton (ed) Machine Learning Methods for Planning. San Mateo, CA: Morgan Kaufmann.