Generalizing and Executing Plans

Christian Muise
Department of Computer Science,
University of Toronto,
Toronto, Ontario, CANADA.
cjmuise@cs.toronto.edu

The Problem

We address the problem of generalizing a plan to maximize the flexibility and robustness with which an agent can execute it. A key area of automated planning is the study of how to generate a plan for an agent to execute. The plan itself may take on many forms: a sequence of actions, a partial ordering over a set of actions, or a procedure-like description of what the agent should do. Many forms can be viewed as a family of plans. The question remains as to how the agent should execute the plan. For simple forms of representation (e.g., a sequence of actions), the answer to this question is straightforward. However, when the representation is more expressive (e.g., a GOLOG program or HTN (Levesque et al. 1997; Erol, Hendler, and Nau 1995)), or the agent is acting in an uncertain world, execution can be considerably more challenging. We focus on generalizing different plan representations before execution begins into a form that allows for nimble and robust execution by an agent.

Proposed Research Plan

Our approach is to generalize the representation of plans to enable an online agent to quickly react to changes in the world. In doing so, we also aim to improve the robustness of the agent by increasing the number of ways it can execute the plan. For example, we compactly represent relevant state information for the partial-order plan (POP) representation. This state information allows an agent to recognize whether or not it can reach the goal with a fragment of the current POP when the agent is faced with an unexpected world state.

We represent the generalized plan as a policy that maps the state of the world to an appropriate action. When faced with a decision, the agent consults the policy for the appropriate action to execute. Using the perspective of a policy to encompass the agent's behaviour allows us to apply our approach to a wide variety of plan forms. The key question for each is how to suitably generalize the plan to create a policy. Using a single representation for the generalized plan provides a unified framework for monitoring the execution of a variety of input plan forms. We consider a range of plan forms for generalization including sequential plans, POPs, Hierarchical Task Networks, and GOLOG programs.

Progress To Date

To date we have focused on generalizing partial-order plans (POPs). We described how to generalize a POP for online execution, and laid the groundwork for the approach our research will follow (Muise, McIlraith, and Beck 2011a). We also presented a method to optimally relax a sequential plan to a POP (Muise, Mcilraith, and Beck 2011b), providing a connection for generalizing sequential plans to our previous work. In this section, we discuss these contributions and situate them in the context of our overall approach.

Historically, executing a POP involves performing the actions at the start of the POP and updating the plan accordingly. When something unexpected occurs, the agent repairs the plan; either through predefined failure mode strategies or replanning (Veloso, Pollack, and Cox 1998; Wilkins 1985). These approaches, however, are restrictive as they do not recognize actions that are no longer required or must be executed again. We remedy these weaknesses through generalization of the POP.

Our generalized representation builds off of an ordered list of condition-action pairs. The condition in a pair represents what must hold in the state of the world in order for a suffix of the plan, starting with the pair's action, to reach the goal. The order of the list represents a preference to execute actions earlier in the list as long as their associated condition matches the state of the world. Once a plan is generalized to an ordered list of condition-action pairs, we build a partial policy that maps the conditions to their associated actions. When more than one condition matches the current state, we prefer those pairs that were found closer to the goal. This behaviour is captured in an incrementally constructed ordered algebraic decision diagram (OADD). The OADD uses the facts in the world as decision nodes and appropriate actions as leaf nodes.

To construct the list of condition-action pairs from a POP, we follow a strategy similar to Fritz and McIlraith (2007): we regress the goal condition through the POP, recording the appropriate action and condition for every possible suffix. Since a POP could potentially represent millions of linearizations, we introduce a systematic method of implicit regression through all the linearizations. We identify and deal with any duplication in the conditions for a set of actions in a suffix. Our approach to regressing through the POP greatly reduces the size of the condition-action list, and allows for a

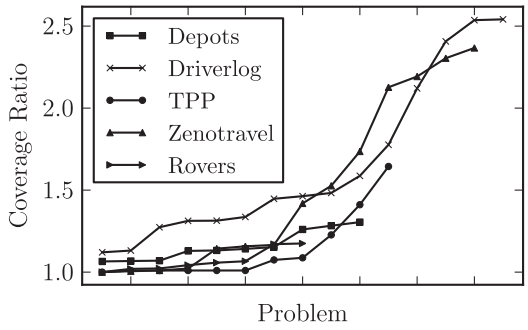

Figure 1: Analytic comparison of state coverage. The y-axis indicates the coverage when using the generalized plan divided by the coverage when using the sequential plan. We sort problems from each domain based on their y-axis value.

compact representation of the partial policy.

Using the constructed policy is extremely efficient, and its strength is largely due to its ability to quickly determine if any condition applies (Muise, McIlraith, and Beck 2011a). The complexity for using the policy to return an action is linear in the height of the OADD (bounded by the number of possible facts in the world). We refer to the number of states for which a policy returns an action for as the *coverage* of the generalized plan. Figure 1 shows the relative increase in coverage of our approach as compared to using just a sequential plan in a number of domains. We also identified several properties of a POP that cause an exponential increase in the coverage of using a POP rather than a sequential plan (Muise, McIlraith, and Beck 2011a).

A sequential plan is just a special case of a POP that has the maximum number of ordering constraints on the actions. To leverage our work on POPs, we developed a method to optimally relax a sequential plan to be a POP with as few actions and ordering constraints as possible (Muise, Mcilraith, and Beck 2011b). We introduced a novel partial weighted MaxSAT encoding and used an existing MaxSAT solver to find a solution.

Previous approaches have been developed to compute a partial-order relaxation of a sequential plan (e.g., (Bäckström 1998)), but they are restricted to relaxing the ordering of the plan only (i.e., they cannot reorder actions in the plan). Using our encoding, we computed POPs that include exponentially more linearizations of the actions in the plan. As a result, the POPs are far more flexible as the added linearizations increase the number of ways that we can execute the actions by up to several orders of magnitude.

Timeline and Future Work

To date we have investigated only sequential plans and POPs, but the generality of our approach opens the door to a wide variety of other solution forms. An approach that exploits our representation of a partial policy to do fully observable non-deterministic planning was submitted to the International Conference on Automated Planning and Scheduling (Muise, McIlraith, and Beck 2012). Currently, we are investigating the possibility of generalizing a richer form of partial-order plan that implicitly represents a family of POPs, and expect to complete this work before May, 2012. Similar to our work on POPs, we have developed a systematic form of regression that implicitly takes advantage of similarities between the various linearizations.

We hope to investigate two further representations that encompass a family of plans (just as a POP encompasses a family of linearizations): Hierarchical Task Networks (Erol, Hendler, and Nau 1995) and GOLOG programs (Levesque et al. 1997). In both cases, the representation itself is not typically considered ready for execution by an agent. To remedy this, we will introduce the notion of *intended effects* for the components in the plan. This will allow us to effectively produce the condition-action list required for our approach to be applicable. We aim to complete this work by October, 2012.

Conclusion

Our research aims to provide three core contributions to the area of automated planning: 1. A formal characterization of the conditions under which a complex plan remains viable, 2. A principled method of generalizing various plan representations, and 3. A unified policy representation that embodies a generalized plan and allows an agent to execute efficiently. We have demonstrated the potential of our approach for sequential and partial-order plans, and we are currently extending our work to a richer form of partial-order plan. In a dynamic environment, an intelligent agent must consider contingencies and plan for them. We aim to address this key issue by building more robust artificial agents through the generalization of a variety of plan forms.

References

Bäckström, C. 1998. Computational aspects of reordering plans. *Journal of Artificial Intelligence Research* 9(1):99–137.

Erol, K.; Hendler, J.; and Nau, D. 1995. Htn planning: Complexity and expressivity. In *Proceedings of the National Conference on Artificial Intelligence*, 1123–1123.

Fritz, C., and McIlraith, S. A. 2007. Monitoring plan optimality during execution. In *Proceedings of the International Conference on Automated Planning and Scheduling*, 144–151.

Levesque, H.; Reiter, R.; Lespérance, Y.; Lin, F.; and Scherl, R. 1997. GOLOG: A logic programming language for dynamic domains. *The Journal of Logic Programming* 31(1-3):59–83.

Muise, C.; McIlraith, S. A.; and Beck, J. C. 2011a. Monitoring the execution of partial-order plans via regression. In *Proceedings of the International Joint Conference On Artificial Intelligence*.

Muise, C.; Mcilraith, S. A.; and Beck, J. C. 2011b. Optimization of partial-order plans via maxsat. In *Workshop on Constraint Satisfaction Techniques for Planning and Scheduling Problems*.

Muise, C.; McIlraith, S. A.; and Beck, J. C. 2012. Improved non-deterministic planning by exploiting state relevance. In *International Conference on Automated Planning and Scheduling*.

Veloso, M.; Pollack, M.; and Cox, M. 1998. Rationale-based monitoring for planning in dynamic environments. In *Proceedings of the International Conference on AI Planning Systems*, 171–179.

Wilkins, D. 1985. Recovering from execution errors in SIPE. *Computational Intelligence* 1(1):33–45.