Hybrid Planning: An Approach to Integrating Generative and Case-Based Planning

Marie desJardins
SRI International
333 Ravenswood Ave.
Menlo Park CA 94025
marie@erg.sri.com

Anthony Francis*
Georgia Institute of Technology
AI/Cognitive Science Group
College of Computing
Atlanta GA 30332-0280
centaur@cc.gatech.edu

Michael Wolverton
SRI International
333 Ravenswood Ave.
Menlo Park CA 94025
mjw@erg.sri.com

Abstract

This paper describes ongoing research on the development of a *hybrid planning* system that integrates case-based reasoning (CBR) methods into SIPE-2,† a generative planning system. Novel aspects of this work include the integration of CBR methods with a hierarchical task network (HTN) planner; the estimation of adaptation cost based on a classification of planning conditions; the reuse of plans using SIPE-2's replanning techniques; the extraction and application of *advice* from retrieved cases to guide planning in a form of plan replay; the representation and use of *opaque constraints*, which describe the human planner's decision factors; and the provision of case-based support for managing a distributed, continuous planning process.

Introduction

We are developing a *hybrid planning* approach that integrates case-based planning methods into SIPE-2, a hierarchical task network (HTN) planner (Wilkins 1988).

This hybrid planning research is being performed as part of an applied research project entitled "Joint Maritime Crisis Action Planning (JMCAP)." Crisis action planning demands quick, continuous, distributed planning. The goal of the JMCAP project is to provide support in this environment for *mixed-initiative* planning, in which the human planner and the automated system jointly create a plan. Crisis action planning, which is currently performed entirely by human planners, is a mixture of first-principles procedures (e.g., Navy and Marine doctrine) and past experience. Providing mixed-initiative planning support therefore requires the integration of first-principles planning methods with case-based methods that permit plan reuse.

In addition to this operational motivation, there are a number of functional reasons to add case-based methods to a generative planner. The first and most straightforward of these is performance speed-up. Next, in a mixed-initiative environment, case-based methods enable the system to learn over time by recording planning decisions made by a human. Finally, by noting and recording decisions and conflicts in the distributed, continuous planning process, the system can learn over time to manage this process more effectively.

The rest of this paper describes our hybrid planning approach. (This research is in its early stages, and the approach is only partially implemented.) We first discuss the natural similarities between SIPE-2's operators and cases in a CBR system, and then describe how cases are represented, created, retrieved, and adapted. Finally, we briefly describe how we propose to represent human decision factors and to support distributed planning, then summarize our conclusions.

Operators and Cases

SIPE-2 uses planning operators (templates describing how to expand goals into subgoals and tasks) to hierarchically decompose high-level goals into low-level (executable) actions. Each operator has a *purpose*, which describes the type of goal that it can be used to expand, a set of *preconditions*, which specify the situations (world state) for which it is relevant, and a *plot* (expansion), which is a partially ordered network of subgoals and actions. An operator is indexed by its purpose, and the preconditions are used to determine whether to apply the operator in the current situation. When an operator is matched, the planner expands the goal by splicing the operator's plot into the plan where the goal appears, and adding appropriate constraints and variable bindings.

We plan to use case-based methods within SIPE-2 by identifying appropriate cases at each level during the hierarchical decomposition process. This will occur at run time, at the same time that appropriate operators are identified. Cases in this system are treated as generalized operators. The key difference between operators and cases is that cases generally will not match the situation exactly, so the indexing, retrieval, and application methods will differ. During planning, the best case or operator will be selected (either by the user, or automatically, using internal evaluation criteria). If an operator is selected, it will be applied as usual. If a case is selected, it must be adapted so as to apply to the current situation. This will be done by means of

*Work performed while at SRI International
†SIPE-2 is a trademark of SRI International

SIPE-2's replanning methods or advice extraction (both described later in the paper).

Case Representation

We have developed a case representation that includes the problem that the case has solved (analogous to the purpose of an operator); a plan or plan fragment (analogous to the plot); the context in which the plan was applied (analogous to the preconditions); and outcome information (e.g., execution-time feedback), which may be used as a factor in case selection.

The core component of a case is the plan it contains. (The plan is the *solution* part of the case, in traditional CBR terminology.) A SIPE-2 plan consists of multiple levels. The top-level plan is the initial problem (the highest-level goal(s) to be achieved). Each level below the top level is a partially ordered network of subgoals and actions that describes an intermediate plan expansion. A goal at level i is linked to its expansion at level $i+1$ through descendant and ancestor links. The expansion, or *plan wedge*, is simply the plot of the operator that was applied to expand the higher-level goal. As the subgoals are expanded, all of the ordering links are maintained, and new links can be introduced by the planner to order actions in parallel branches (e.g., to enable the effect of an action introduced by one operator to satisfy a precondition required by another).

The context of the case consists of the conditions in the current situation that are relevant to the application of the cases, as discussed in the next section; and the human decision factors associated with the plan, which are discussed below ("Plan Rationale").

Case Indexing

The hybrid planner must provide efficient mechanisms for storing plans as cases in the libraries, and indexing them to facilitate their retrieval and reuse. In particular, the system must identify features (such as objectives and constraints) for indexing the stored plans.

Recent results in case-based reasoning research suggest that ease of adaptation is a useful similarity metric (Smyth & Keane 1995). Projecting the adaptation cost of a case is in general a difficult problem, but SIPE-2's rich plan representation helps us to do this. Embedded in the plan structure is a record of how the planning operators were applied to produce the plan; the operators themselves encode information about their applicability. We identify relevant preconditions on which the original plan depends, then classify these into *condition classes* according to the expected cost of violating them, as discussed in the next section.

To determine these relevant preconditions, we have extended PRODIGY/Analogy's footprint indexing method (Veloso 1995). The footprint of a goal or problem is the subset of the initial world state that has contributed to the solution of the goal—that is, the conditions that have made a solution possible, or influenced the structure of a solution. In STRIPS-based planners like PRODIGY/Analogy, computing the footprint of a goal involves regression from the goal back through the preconditions of the operator network to the literals of the initial world state. In hierarchical planners like SIPE-2, computing the footprint is similar, but requires a slightly more complex process to analyze the plan structure below the goal (i.e., the goal's expansion). (A detailed description of this algorithm is beyond the scope of this paper.)

Footprint Condition Classes

SIPE-2 splits the necessary conditions for an operator's application into two sets: the subgoals that it must try to achieve, and the preconditions that make it valid. SIPE-2 will not try to make a precondition true: if an operator's precondition cannot be satisfied (by a predicate in the initial world state, or by an effect of an action that appears earlier in the plan), the operator is invalid and will not be considered for application.

We can segregate the footprint of a goal into conditions that support subgoals, and conditions that support the operator preconditions. For plan adaptation in SIPE-2, precondition support is more important than goal support. If a condition that supported a subgoal is missing in a new situation, the rest of the plan is not necessarily invalidated: an adaptation process can likely replan for the open goal without disturbing the remainder of the plan. But if a condition that supported a precondition is missing, that operator is invalid, and the entire plan wedge corresponding to the operator's expansion must be removed.

We have developed a set of condition classes that can be used to further distinguish among types of precondition support. These condition classes can be understood by considering the operators that were used to generate the plan. First, a precondition may be unique to an operator and incompatible with the preconditions of other operators. For example, in the maritime planning domain, there might be several operators that describe strategies to perform noncombatant evacuation, each tailored to a specific type of terrain. In this case, changing the terrain would force us to select another operator. We refer to a condition that satisfies this type of precondition as a *hinge*: the operator choice is effectively determined by the presence or absence of the condition.

Another important situation arises when all operators that can achieve a goal depend on a single shared precondition: for example, all operators that achieve the goal to evacuate noncombatants on rigid inflatable boats (RIBs) depend on the weather being appropriate (the waters must be calm). If this condition is not satisfied, the planner will not be able to find a new operator to achieve this goal—the goal itself, along with the operator that introduced it at the higher level, must be replanned. We refer to this kind of initial condition as a *linchpin*: removing it makes higher-level portions of the plan fall away.

In summary, we distinguish the following qualitative classes of conditions in the initial world state of a plan, listed in order of increasing importance (or, more accurately, in order of increasing cost of repairing a plan when they are violated):

- **External:** a condition in the initial world state that does not support a precondition or goal of the plan. Removing such a condition will not change the formal properties of the plan (although the condition may be highly significant to the user).
- **Phantomizing (or Goal):** a condition that phantomizes a goal, and therefore can likely be replanned for if removed.
- **Enabling (or Precondition):** a condition that makes an operator precondition true, and that would invalidate a plan wedge if removed.
- **Hinge:** a subclass of preconditions that determine an operator choice—i.e., that discriminate among several operators that solve the goal.
- **Linchpin:** a subclass of preconditions that uniquely allow us to satisfy a particular goal—i.e., all operator choices that can satisfy this goal require this precondition.

We expect that the actual cost of violating a given class of precondition will be largely domain dependent. (Costs include both computational costs and time delays in the planning cycle.) We plan to apply adaptive techniques to learn these costs empirically.

Case Retrieval

Retrieving cases from a library requires that the current and past problems and situations be compared, to determine which cases are most similar or most applicable. The hybrid planner will use the footprint condition classes described above to index plans in the library. The planner will also retrieve plans that are estimated to require the least adaptation cost. This estimation will be based on the expected violation costs of the conditions in the retrieved case that fail to match the current situation.

Other retrieval criteria we are investigating include the use of domain-specific knowledge to compute a weighted match (e.g., matching higher-level objectives and operational constraints is more important than matching low-level details such as available resource types); additional plan analysis to estimate the cost of replanning to adapt the retrieved plan; and extracting advice before storing a case to use as additional indices (see "Plan Replay—Advisable Planning").

Case Adaptation

Once a case has been retrieved, it must be applied, using the differences between the initial situation and the current situation to adapt the solution. We are exploring two approaches to case adaptation: reuse and replay. In case reuse, the case consists of a description of a problem, and an associated solution, which is retrieved and spliced into the plan. In case replay, on the other hand, a trace of the original planning *process* is stored and used to solve the new problem by guiding the search, essentially creating an analogy of the previous solution in the current context. SIPE-2's plan structure already records all of the planning decisions and constraints, so only minimal extensions to the representation are required to support both plan reuse and plan replay.

We are extending existing SIPE-2 components to provide these two approaches to case adaptation, and will then explore heuristic methods for deciding which of the two approaches is appropriate for any given case retrieval. SIPE-2's replanner serves as the core for the plan reuse system, and the Advisable Planner is the foundation for plan replay. These are described in the following subsections.

Plan Reuse—Replanning in SIPE-2

Plan reuse in SIPE-2 entails adjusting the initial conditions of the retrieved plan to match the new situation, eliminating invalid or irrelevant parts of the retrieved plan, adding new goals that may have been absent in the original plan, and eliminating the goals in the case that are not part of the new problem.

Adjusting the initial conditions requires matching the objects and predicates in the retrieved case to the current situation. SIPE-2's replanner already does part of this work, but it has only a limited ability to rebind variables, so we are extending the replanner to do more sophisticated variable matching and rebinding.

Plan critics within SIPE-2 are used during both planning and replanning to evaluate the quality of partial or complete plans along multiple dimensions (e.g., resource utilization, temporal constraints, and interaction of parallel actions). The replanner applies the plan critics to an existing plan (the retrieved case), using an updated situation description (the current situation), and detects failed preconditions and phantoms that are no longer valid. These failures cause wedges of the plan to be spliced out and replanned.

The hybrid planner can additionally introduce new top-level goals that did not appear in the retrieved plan, or delete goals that are not relevant in the current situation. The new goals are incorporated into the plan network at the top level, and expanded via the normal plan expansion methods (i.e., operator expansion or the use of additional cases). Deleted goals are removed from the plan network, which entails removing the plan wedges corresponding to the goals' expansions. After a goal is deleted, the replanner again applies the critics to determine whether other parts of the plan depended on conditions produced in this wedge.

Unfortunately, in some circumstances the replanner may eliminate more of the plan than necessary: for example, a high-level operator might be invalidated, even though most of the wedge beneath it would remain valid once replanning was complete. The replanner cannot

preserve this valuable past planning knowledge; this is one of the cases where the system can apply the plan replay methods we introduce next.

Plan Replay—Advisable Planning

The Advisable Planner (AP) was originally developed to support mixed-initiative planning by allowing a user to specify constraints on a plan in the form of *advice* (Myers 1996). AP implements three classes of advice: role advice (what objects should fill certain variables in a plan), method advice (what types of operators should be used to solve certain goals), and sketch advice (what specific operators should be used to solve certain goals under certain conditions).

We are using AP to develop a plan replay mechanism that is more flexible than traditional replay. Advice will be extracted from a plan and used for case indexing, retrieval, and adaptation (by applying the advice in the new planning process). One straightforward approach to advice extraction is to use each operator application in the plan to encode a piece of sketch advice. More sophisticated approaches could be used to detect the objects or classes of operations used in the plan. We also expect to identify new advice types (e.g., representing a decision to defer the expansion of a particular goal) that would be useful in case adaptation.

Extracting advice from retrieved plans provides a natural way to handle multiple-goal problems not addressed by any single case in the case base. Composing the advice extracted from two single-goal plans permits the planner to be guided by both plans simultaneously. There is a potential for conflicts or redundancy, but SIPE-2's plan critics will identify and resolve these problems.

Plan Rationale

In the current implementation, retrieved cases are assumed to be complete SIPE-2 plan structures. We plan to relax this assumption by developing methods to incorporate incomplete plan structures that are provided by a plan authoring system or another external source. These plans will most likely be missing important information, such as decision points, preconditions, and a detailed situation description. To support the use of these external plans, we are developing a language for annotating a plan structure. This language will include decision factors used by a human planner that are outside SIPE-2's automated reasoning capabilities.

We propose to model human decisions as *opaque constraints*—that is, the system will record that the human planner has made a decision, or indicated that a property of the situation was important, but that the system does not necessarily know why. While the system may not be able to fully reason about these opaque constraints, it will be able to partially determine when they are violated, and use that partial information to interactively (with user support) resolve the conflict, in order to adapt a retrieved plan appropriately.

Distributed Planning

One of the major objectives of the JMCAP project is to develop techniques for distributed, continuous planning. To this end, we have developed a distributed version of SIPE-2 (desJardins & Wolverton 1998; Wolverton & desJardins 1998) in which multiple communicating planning agents cooperate to develop an integrated, consistent joint plan. We envision two ways in which hybrid planning will support the distributed aspects of our planning system.

First, cases can support the process of assigning subgoals to the agents who will solve them. Whenever a planning agent generates a new subgoal, it makes the decision whether to solve the subgoal itself or to assign the subgoal to another agent that it thinks can solve it. By extending our case representation to include successful past goal-to-agent assignments, CBR can support the planning agent's decision by allowing the system to allocate similar goals to similar agents in similar future situations. This support is analogous to the way that cases store successful operator applications: in both situations, CBR selects a method of solving a goal that has a history of success in similar situations, or that a human operator had reason to select; the only difference is that the "method of solving a goal" now is the assignment of that goal to another agent instead of a planning operator.

Second, cases can be used to store extracted advice at the metalevel in order to avoid and resolve conflicts during distributed planning. Planning successes and failures, along with any conflict resolution actions taken, will be recorded and stored with the case, and will be converted into advice to guide the new planning process. This recorded information will include backtracking points and reasons, plan repairs performed at execution time in response to execution failures or opportunistic goal achievement, and conflicts that arise during distributed planning or plan merging. These retrieved cases will be used to guide the distributed planning process, by suggesting the information requirements of distributed agents and the planning choices that are most likely to minimize conflicts.

Conclusions

We have described a hybrid planning approach that integrates case-based methods into a generative planning system (SIPE-2). This work is preliminary, but represents a significant advance in the state of the art of case-based and generative planning integration.

Acknowledgments

This research was funded by the Office of Naval Research (ONR) and the Space and Naval Warfare Systems Command (SPAWAR) under contract N66001-97-C-8515. The authors would like to thank Mr. Dave Swanson, JMCAP Project Manager at SPAWAR, for inputs and feedback. Thanks also to Karen Myers and

David Wilkins of SRI International for their contributions to this work.

References

desJardins, M., and Wolverton, M. J. 1998. Coordinating planning activity and information flow in a distributed planning system. In desJardins, M., ed., *AAAI Fall Symposium on Distributed Continual Planning.* AAAI Press Technical Report (forthcoming).

Myers, K. L. 1996. Advisable planning systems. In Tate, A., ed., *Advanced Planning Technology.* AAAI Press.

Smyth, B., and Keane, M. 1995. Experiments on adaptation-guided retrieval in a case-based design system. In Veloso, M., and Aamodt, A., eds., *Case-Based Reasoning Research and Development.* New York: Springer Verlag. 313–324.

Veloso, M. M. 1995. *Planning and Learning by Analogical Reasoning.* Springer-Verlag.

Wilkins, D. E. 1988. *Practical Planning: Extending the Classical AI Planning Paradigm.* Morgan Kaufmann.

Wolverton, M. J., and desJardins, M. 1998. Controlling communication in distributed planning using irrelevance reasoning. In *AAAI-98.* To appear.

Appendix

1. **Integration name/category**: JMCAP hybrid planning
2. **Performance task**: Hierarchical plan generation in a military (maritime planning) domain
3. **Integration objective**: Efficiency (computational and user interaction), reuse of expert decision-making criteria
4. **Reasoning components**: Plan retrieval, constraint-based plan adaptation (repair), advice extraction for plan reuse, plan merging, footprint extraction (indexing)
5. **Control architecture**: Primarily CBR as slave
6. **CBR cycle step(s) supported**: Retrieval, Reuse, Revision, Merging, Retention
7. **Representations**: Plans, advice
8. **Additional components:** User interaction (during retrieval and repair)
9. **Integration status:** Proposed
10. **Priority future work:** Implementation, development of case library, empirical evaluation.