

Prioritizing Planning Decisions in Real-World Plan Authoring

Michael Wolverton

Artificial Intelligence Center
SRI International
333 Ravenswood Ave
Menlo Park, California 94025
mjw@ai.sri.com

Abstract

One promising direction for moving AI planning into the real world is to build systems that are more *user-centric*, in that they allow the user—not the system—to make many of the decisions necessary to create the final plan. However, since there are a large number of decisions to make in the course of producing a plan, shifting the responsibility for those decisions to the human planner runs the risk of overwhelming the human planner with too many choices. One approach to helping the human planner manage the large number of decisions is to automatically *prioritize* those decisions according to their importance or urgency in the current planning context. This paper describes two methods for automatically prioritizing planning decisions. One is a *commitment-based* approach, which prioritizes decisions according to the number of future decisions they eliminate from the planning process. The other is an *experience-based* approach, which prioritizes decisions according to the order in which they have been performed in previous planning sessions. Both approaches have been implemented in PASSAT, a plan-authoring system in which users construct and modify plans interactively using a library of templates.

Introduction

A major obstacle to the widespread acceptance and use of Artificial Intelligence planning systems is the lack of control afforded by those systems. The systems have traditionally been designed as black boxes: they take a description of a domain and a set of goals and automatically synthesize a plan for achieving the goals. Human planners, however, are generally reluctant to cede full control of their plan.

One promising direction for moving AI planning into the real world is to build systems that are more *user-centric*, in that they allow the user—not the system—to make many of the decisions necessary to create the final plan. However, while user-centric planning systems may solve the problem of lack of user control, they introduce a new problem: there are a lot of decisions to make in the course of producing a plan. At any given point in the creation of a complex plan, there may be hundreds or even thousands of unresolved issues. Shifting the responsibility for those decisions (or even

a substantial subset of them) from the system to the human planner runs the risk of overwhelming the human planner with too many choices.

One component of a solution to this problem is to keep the human planner focused on the most important tasks. To do this, we need some way of estimating the relative importance of the various planning tasks facing the user in the current planning context.

This paper describes two methods for automatically prioritizing planning decisions. One is a *commitment-based* approach, which prioritizes tasks according to the number of future decisions they eliminate from the planning process. The other is an *experience-based* approach, which prioritizes tasks according to the order in which they have been performed in previous planning sessions. Both approaches have been implemented in PASSAT (Myers *et al.* 2002), a plan-authoring system in which users construct and modify plans interactively using a library of *templates*.

The remainder of the paper is organized as follows. First, we describe the PASSAT system and its agenda of planning steps. Next, we introduce the prioritization issue in more detail and discuss some possible criteria for prioritization. Then we describe and discuss the commitment-based and experience-based prioritization methods and their implementation within PASSAT. Finally, we conclude by discussing future directions for this ongoing research.

PASSAT

Figure 1 shows a snapshot of the PASSAT interface during a planning session. The large frame on the left contains a hierarchical decomposition of the current partial plan. Items next to folder icons are tasks that have been expanded; items next to star icons are tasks that can be expanded further (either through automated template application or interactively); and items next to document icons are tasks that match no templates. The frame on the upper right shows the current agenda—the list of planning steps the user must perform to address outstanding issues. The frame on the lower right shows the list of information requirements—sources of information that have been identified by the user or PASSAT's planning knowledge as relevant to various portions of the planning process.

The human planner develops the plan by selecting a planning step from the agenda and performing that step (many of

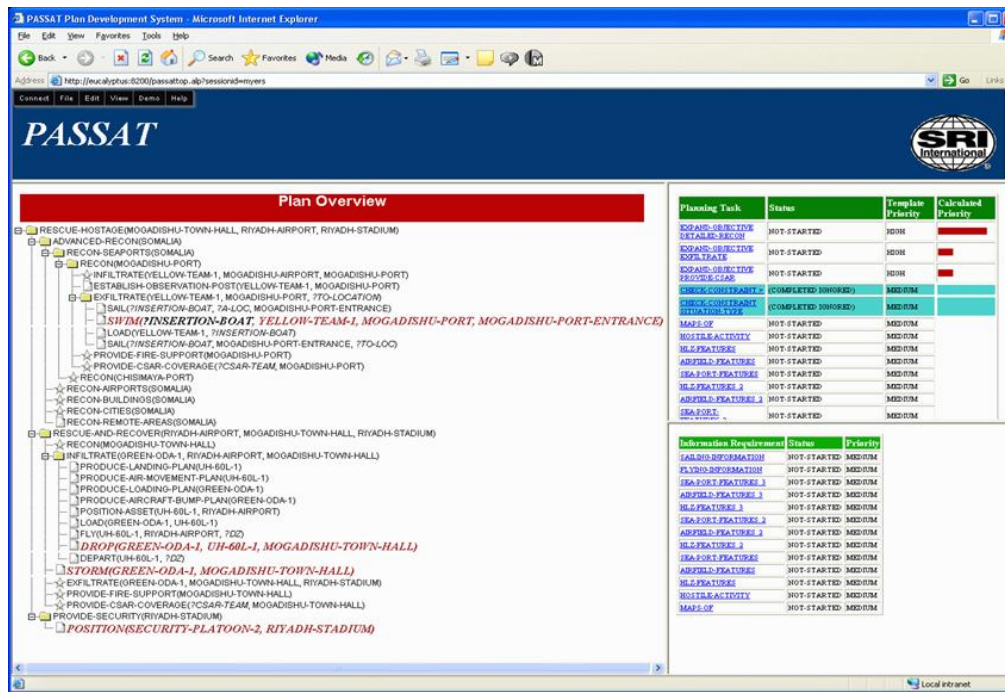


Figure 1: PASSAT Interface

these planning steps are accessible through the plan display as well). If the planning step is to expand the PROVIDE-CSAR-COVERAGE task, for example, the planner would be presented with several options: apply one of the templates that matches the task, drop the task, or create a sketch for achieving the PROVIDE-CSAR-COVERAGE task and work with PASSAT to refine that sketch. Performing this planning step may cause additional planning steps to be added to the agenda (i.e., new tasks, variables, and constraints may have been introduced into the plan) and new information requirements as well.

Plan Representation

PASSAT's representation of plans and tasks is based on a fairly standard HTN model (similar to that of (Erol, Hendler, & Nau 1994)), augmented with a rich temporal representation for tasks. Using PASSAT, a user would describe the objective of the plan in the form of one or more task statements, each consisting of a task operator and terms (variables, instances, or functions applied to terms).

Templates A template describes one way that a task (i.e., the template's purpose) can be decomposed into subtasks. A template consists of a set of these subtasks, as well the variables used in the template, constraints on the applicability of the template, and the effects of successfully performing individual tasks and the entire template. Different templates may describe different decompositions for the same task.

Constraints Constraints consist of state predicates that denote hard or soft conditions, perhaps due to physical laws or policy rules. PASSAT employs a three-valued logic for

constraints, grounded in the values TRUE, FALSE, and UNKNOWN. Unlike in automated planning systems, a constraint with value other than TRUE does not necessarily halt the process or cause backtracking. Instead, a violated constraint is called to the attention of the user, who has the choice of ignoring the violation or changing the step that triggered the violation.

User-centric Plan Development

PASSAT provides two main modes of plan development: interactive plan refinement and plan sketching. Because the prioritization techniques that are the subject of this paper can be understood without knowledge of plan sketching, we will only describe interactive plan refinement here. See (Myers *et al.* 2003) for a detailed description of PASSAT's plan sketching capability.

Interactive plan refinement in PASSAT involves three types of planning step: expand task, instantiate variable, and resolve constraint.

Expand Task When a task is to be expanded, the system offers the user the choice of applying a predefined template, sketching a solution, or dropping the task.

When the user chooses a template to apply, the system first unifies the task and the template's purpose, making appropriate substitutions throughout the template. PASSAT adds the (partially instantiated) subtasks and constraints of the template to the plan. In addition, it extends the agenda to include planning steps to expand the new subtasks, to check the new constraints, and to instantiate any unbound variables from the template. The planning step for the parent task is

marked as completed and removed from the agenda. In the displayed plan, the parent task is shown with its subtasks.

As the system performs this step, it also checks the status of all new constraints. If one is found to be valid, the planning step to check it is marked as completed and removed from the agenda. If it is found to be invalid, the planning step is flagged.

As the system expands a task, other planning steps may be affected. If the unification results in the assignment of a value to a variable, the planning step for instantiating that variable is removed. The status of constraints that contained that variable might now be resolvable; the system checks those constraints and updates the planning steps, if necessary.

Instantiate Variable The agenda contains a planning step for each unbound variable within the current plan. When the user is ready to instantiate a variable, PASSAT provides the set of possible instantiations that satisfy all relevant constraints; the user can select from this set, provide an alternative value (hence, overriding a relevant constraint), or simply mark some subset of the values as unacceptable. When the variable is instantiated, any impacted constraints are rechecked. A user can optionally provide a justification (currently, a text string) for his actions.

Resolve Constraint As noted above, PASSAT provides automated checking of constraints as part of template application, with the agenda being used to track constraints that the system was unable to validate. Resolve constraint steps enable a user to declare that the system can disregard individual constraints with the status of FALSE or UNKNOWN in a given situation. Such declarations do not have assertional import (i.e., they do not change the system's world model); rather, they enable relaxation of constraints from the planning model embodied in the domain templates. A user can declare that a given constraint be ignored for a variety of reasons: (a) he has more recent information that would validate the constraint, (b) he knows that the constraint is overly strong for the current situation, or (c) he wants to explore a what-if scenario. PASSAT supports the user in providing a justification (currently, a text string) for such constraint relaxations.

Process Facilitation in PASSAT

While the PASSAT system is not designed to make critical decisions for the human planner, it is designed to offer guidance on the user's decisions during the planning process. PASSAT facilitates the user's plan authoring process by helping the user keep track of open tasks and other information important for the development of the plan. Process facilitation is supported primarily by two capabilities:

- An *agenda* of planning steps listing the decisions the user must make in order to complete the plan. By "planning steps," we mean decisions and actions that the user makes in the process of developing the plan; these are distinguished from "execution tasks," the activities that are part of the plan itself. Each of the agenda's planning steps are of one of the three types described above—Expand

Objective, Instantiate Variable, and Resolve Constraint. The planning steps PASSAT displays in its agenda can be filtered by the user along several dimensions, e.g., task type(s) and completion status.

- A mechanism for identifying key *information requirements* implicit in the user's partial plan, and for directing the user's attention to relevant plan elements when new information arrives.

The agenda is the object of our prioritization work, and we now describe the mechanisms PASSAT uses to order the planning steps on that agenda.

Motivation

In this section, we briefly motivate the need for and introduce some of the issues in planning step prioritization.

In any realistically complex planning domain, the development of a plan can involve hundreds or even thousands of decisions. Correspondingly, PASSAT's agenda can grow quite long in the middle of the planning process. The system provides some basic mechanisms to control agenda growth—instantiating variables during template application, automatic calculation of constraints—and to control information overload in the agenda display—agenda filtering and sorting. However, even with these capabilities, the agenda can frequently reach a size that is overwhelming to the user. In the face of a large number of planning steps, we need a technique for keeping the human planner focused on the most important ones.

To deal with this problem, we have investigated mechanisms for prioritizing the planning steps on the agenda, according to some notion of a task's importance to the planning process. There are a number of possible instantiations of "importance" here, including:

- Commitment-based: Prefer decisions that reduce uncertainty in the planning process.
- Experience-based: Prefer decisions that have been preferred in similar situations in the past.
- Knowledge-based: Prefer decisions that the system's planning knowledge identifies as important.
- Deadline/urgency-based: Prefer decisions that involve execution tasks that are scheduled to start soon.
- Backtracking-based: Prefer decisions that are difficult to achieve. This is effectively the prioritization criterion of the Fewest Alternatives First strategy and related heuristics (Pollack, Joslin, & Paolucci 1997) used in automated planning.

The first three of these approaches have been implemented in PASSAT. The commitment-based and experience-based approaches are described in the next two sections. For the knowledge-based approach, PASSAT allows templates to specify a priority for planning steps. Each subtask, variable, and constraint in a template may be tagged with a qualitative priority (HIGH, MEDIUM, or LOW), corresponding to the importance of making a decision about that entity (expanding the objective, instantiating the variable, checking the constraint). Template-defined priorities always take

precedence over PASSAT’s other prioritization methods in ordering the agenda display.

Commitment-based prioritization

PASSAT’s Commitment-based prioritization technique is designed to focus the planning process on tasks that in turn focus the plan. At an abstract level, the aim of the technique is to direct the user to perform high level actions that will commit the plan toward specific directions and away from others—in other words, those actions that will reduce the plan space as much as possible. At a concrete level, this technique prefers planning steps based on their ability to eliminate future possible planning decisions.

This is a particularly appropriate prioritization strategy for *distributed* planning and plan authoring, where the various planning objectives may be divided among multiple planning nodes. In distributed planning situations, a given planning node may be dependent on other nodes to establish conditions it depends on, or those conditions may be threatened by actions other nodes may place in the plan. For examples of these kinds of node interactions, see (Corkhill 1979) or (Wolverton & desJardins 1998). Narrowing down the planning possibilities as early in the process as possible helps each node know which of its own decisions do and (especially) do not have potential interactions with the decisions of other nodes.

To prioritize planning steps according to their level of commitment, we want to estimate, for each open step, the number of possible future decisions that performing that step will eliminate. Here, we use the term *decision* to refer to a choice point that involves more than one alternative. This is distinct from a *choice*, which is the selection of a particular alternative. A decision can either be an objective for which the planner is choosing a template, or a variable for which the planner is choosing an instantiation. Both the discussion below and the current implementation within PASSAT cover only the former (objective/template decisions); the principle of eliminating future decisions applies to variables as well, although there are some practical problems with extending the calculation to variable decisions that we discuss below.

Note that this prioritization technique is concerned with estimating the number of *possible* decisions that a step eliminates, not the *expected* number of decisions. Since we are prioritizing decisions without knowing the exact choice the human planner is going to make, the number of expected decisions a task will eliminate will always be the same for every task—exactly one.

Also note that the estimate of possible decisions eliminated assumes no backtracking. While PASSAT does support backtracking (via an UNDO option), we expect backtracking to occur far less often in PASSAT than it does in an automated planning system. This expectation is based on the assumption that the human planner will usually have a strong idea of what objectives and actions need to be included in the final plan, and therefore will have a very good (if not perfect) search control heuristic for selecting templates. So even if performing a given planning step technically doesn’t eliminate any possible decisions, since the user could backtrack to any of the “eliminated” decisions, it

is reasonable to think of rejected alternatives (and their descendants) as “eliminated” for the purposes of an estimate for prioritization.

To estimate the number of possible future decisions eliminated by a given planning step to expand the objective G , we need to know two values: the total number of decisions involved in planning G before expanding it, and the expected number of decisions after expanding it. The number of decisions eliminated, then, is just the difference between those two values. That is, to get the number of decisions eliminated by expanding G , we count the number of choice points in the decision tree below G and subtract the average number of choice points below G ’s successors. To do that, we build a structure akin to an *operator graph* (Smith & Peot 1993)—more accurately, we remove recursion to create an operator tree—and walk down the tree counting decisions. Since we don’t know which choice the human planner will make to expand G , we assume that each of the choices is equally likely.

Let $Tem(G)$ be the set of templates that match objective G (i.e., the set of templates for which G unifies with its purpose). To cut off recursion, we’ll define a slightly modified set, $\mathcal{T}(G, TS)$, where TS is the sequence of templates whose application resulted in the objective G :

$$\mathcal{T}(G, TS) = \{T \in Tem(G) \mid T \notin TS\}$$

It is necessary to make the simplifying assumption of a limit to the recursive application of templates because there are technically an infinite number of possible decisions if recursion is allowed. The definition above assumes only one application of a template below any given objective in the operator tree, but it could easily be adapted to allow N applications, for some chosen N .¹

Let $P(T, G)$ be the set of objectives produced by applying template T to objective G . Then the total number of possible decisions made in the course of planning for objective G produced by the template sequence TS —designated $\mathcal{D}(G, TS)$ —is defined recursively as the sum of the decisions from all the goals resulting from all the possible templates that match G , plus one more decision if more than one template matches G . The formula for this is given in Equation 1 of Figure 2.

To get the number of decisions left after G has been expanded, we compute the mean number of decisions for all of G ’s successors—designated μ_G —where a successor is the collection of objectives that result from applying a matching template to G . Since the total number of decisions for G ’s successors is the total number of decisions for G minus 1, we just need to divide that number by the number of G ’s successors—i.e., the number of templates that match G . The formula for μ_G is given in Equation 2 of Figure 2.

Then the number of possible decisions eliminated by expanding objective G is the difference between the total decisions for G and the expected total decisions left after G has been expanded one level, plus one (the decision made in the course of expanding G):

¹This assumption is for the purposes of the priority estimate only; PASSAT puts no limit on the number of recursive applications that the planner may choose.

$$\mathcal{D}(G, TS) = \begin{cases} 0, & \mathcal{T}(G, TS) = \phi \\ \sum_{T \in \mathcal{T}(G, TS)} \sum_{G' \in P(T, G)} \mathcal{D}(G', TS \cup \{T\}), & |\mathcal{T}(G, TS)| = 1 \\ 1 + \sum_{T \in \mathcal{T}(G, TS)} \sum_{G' \in P(T, G)} \mathcal{D}(G', TS \cup \{T\}), & \text{otherwise} \end{cases} \quad (1)$$

$$\mu_G = \begin{cases} \frac{\mathcal{D}(G, \phi) - 1}{|\mathcal{T}em(G)|}, & |\mathcal{T}em(G)| > 1 \\ \text{undefined}, & \text{otherwise} \end{cases} \quad (2)$$

Figure 2: Formulae for estimating (1) the number of decisions that could arise during the process of planning objective G , and (2) the average number of possible decisions for the successors of G .

Planning Task	Status	Template Priority	Calculated Priority
EXPAND-OBJECTIVE EVACUATE 2	NOT-STARTED	MEDIUM	<div style="width: 90%; height: 10px; background-color: red;"></div>
EXPAND-OBJECTIVE EVACUATE 3	NOT-STARTED	MEDIUM	<div style="width: 85%; height: 10px; background-color: red;"></div>
EXPAND-OBJECTIVE PSYCHE SUPPORT	NOT-STARTED	MEDIUM	<div style="width: 20%; height: 10px; background-color: red;"></div>
EXPAND-OBJECTIVE PREPARE EVACUATION	NOT-STARTED	MEDIUM	<div style="width: 15%; height: 10px; background-color: red;"></div>
EXPAND-OBJECTIVE RECON-LOCATION	NOT-STARTED	MEDIUM	<div style="width: 10%; height: 10px; background-color: red;"></div>
EXPAND-OBJECTIVE MAIN MOVE	NOT-STARTED	MEDIUM	<div style="width: 10%; height: 10px; background-color: red;"></div>
EXPAND-OBJECTIVE PROVIDE CARE COVERAGE	NOT-STARTED	MEDIUM	<div style="width: 10%; height: 10px; background-color: red;"></div>

Figure 3: Objectives Prioritized in PASSAT

$$Priority(G) = \begin{cases} 0, & |\mathcal{T}em(G)| \leq 1 \\ \mathcal{D}(G, \phi) - \mu_G + 1, & \text{otherwise} \end{cases}$$

Within PASSAT, when the user turns on commitment-based prioritization, this formula is used to order the Expand-Objective planning steps on the agenda. The (normalized) priorities are also shown in a bar chart to the right of the planning steps in the agenda display—see Figure 3. This bar chart gives the user a sense of the relative importance of the various tasks, in addition to the rank ordering.

The major shortcoming of this approach is that it is computationally expensive. Building the operator tree is a process that is exponential in the number of matching templates per objective, and it involves unifying a task statement to multiple template purpose statements at every branch point. PASSAT limits the prioritization to Expand-Objective planning steps only, and that results in tolerable performance. But adding Instantiate-Variable planning steps to the process would make the whole prioritization procedure impossibly slow. (And that is not even considering the fact that there are an infinite number of possible values for many variables.) This points out the need for an approximation technique: a method for estimating the number of decisions below a goal without counting them up explicitly.

Experience-based prioritization

The commitment-based prioritization approach orders the planning agenda according to what the user *should* do, based on a simple (arguably simplistic) notion of “should”: theoretically, the human planner should make the decisions that eliminate the most uncertainty early on in the planning process. An alternate approach, one that recognizes that real-world planning is often more complex than a simple theoretical model, is to order the agenda according to what the user *will* do—that is, to exploit the expertise of the human users of the system by ordering planning steps according to how they were performed in the past. This is the basic aim of PASSAT’s experience-based prioritization technique.

Experience-based prioritization learns an ordering for the current agenda based on preferences demonstrated by past users of the system. Our implementation uses the algorithm of (Cohen, Schapire, & Singer 1999), which learns an ordering in two steps. First, it learns a preference function that returns a level of certainty that one item should be ranked ahead of another. Second, it uses a greedy algorithm to produce an ordering that agrees as much as possible with the pairwise preference function.

In PASSAT, whenever the human planner selects and executes a planning step S from the agenda, the system records a sample preference for S over every other step on the agenda. These sample preferences can be contradictory—i.e., there can be a preference for S_1 over S_2 and another for S_2 over S_1 . Cohen et al.’s algorithm then induces from these sample preferences a general preference function $PREF(S_1, S_2)$ for every pair of planning steps S_1 and S_2 , and then uses the preference function to order the agenda.

One obvious issue is the level of generality of the induction. When the user elects to instantiate the variable HOSTAGE-LOCATION before expanding the objective EVACUATE-HOSTAGES, for example, what exactly should the system assume about the preference the user is expressing? Is it a general desire to always instantiate variables before expanding objectives? Or is it a desire to instantiate the specific variable HOSTAGE-LOCATION before expanding the specific objective EVACUATE-HOSTAGES?

And should the current planning context—the current partial plan, the decisions that have led to the planning steps in question, etc.—play a role in the prioritization? The current implementation in PASSAT takes a simple approach to this issue: it assumes the user’s selections reflect preferences about specific goals and variables, and it ignores the rest of the planning context in learning its preferences.

There are several advantages of this approach to prioritization over the commitment-based approach described in the previous section. One is that it is possible to order all steps on the agenda, not just the Expand-Objective steps. Another is that it exploits the best possible source of domain expertise for prioritizing tasks in that domain: human planners. Human planners will have reasons for preferring one planning step over another that are far more sophisticated than anything we can capture in a simple prioritization theory; this approach allows that sophisticated thinking to drive the system’s prioritization of tasks. On the other hand, one drawback of the approach is that it is difficult to get the training instances needed to learn the ordering. It requires a human planner to develop (potentially many) plans in a given domain without prioritization—a challenge if the agenda grows to hundreds or thousands of planning steps during the course of planning.

Discussion and Future Directions

The vast majority of research in AI planning has focused on automated planning tools, while plan authoring remains relatively understudied. As such, there are many important topics for future work, in user-centric planning generally and planning prioritization specifically. One critical topic for future work in our research is effectively evaluating process facilitation and prioritization techniques for plan authoring tools. Our ultimate aim is for the technology to produce improvement in the human planning process—better plans, faster plan development cycle, rapid plan repair, less human effort in plan production, etc.—and we would like to design experiments that measure that improvement. The most straightforward way to do those measurements is with human subject studies, but designing and carrying out meaningful large-scale human subject studies in realistic planning domains present all kinds of scientific and logistical difficulties.

Another important area for future study is understanding the relationship between prioritization of human planning decisions and search control techniques for automated planners—e.g., (Nau, Smith, & Erol 1998; Tang & Mali 2003; Hoffmann & Nebel 2001; Pollack, Joslin, & Paolucci 1997). One important difference is that automated search control is generally focused on reducing the likelihood of backtracking while moving the search closer to the final optimal solution. In plan authoring, by contrast, we can often assume that the human planner will provide that sort of search control himself: he knows what criteria the final plan should meet, and has the domain expertise to select templates that will push the plan toward those criteria. Prioritization in plan authoring has different purposes: keeping the user focused, minimizing harmful interactions with

other distributed planners, planning urgent objectives first, and others.

Finally, the two prioritization methods described in this paper (plus the template-defined prioritization we mentioned briefly) clearly do not represent the only useful techniques for ordering planning decisions. Eventually, we would like PASSAT to support a large suite of prioritization techniques from which the user can select. Some of the other techniques we envision for inclusion in that suite were discussed in the Motivation section.

Acknowledgements

This research was supported by the Defense Advanced Research Projects Agency (DARPA) under Air Force Research Laboratory (AFRL) contract number F30602-00-C-0058. Thanks to the other members of the PASSAT team—Karen Myers, Mabry Tyson, and Peter Jarvis—for their work on the system and for many helpful discussions on this research.

References

- Cohen, W.; Schapire, R.; and Singer, Y. 1999. Learning to order things. *Journal of Artificial Intelligence Research* 10:243–270.
- Corkhill, D. 1979. Hierarchical planning in a distributed environment. In *IJCAI-79*.
- Erol, K.; Hendler, J.; and Nau, D. 1994. Semantics for hierarchical task-network planning. Technical Report CS-TR-3239, Computer Science Department, University of Maryland.
- Hoffmann, J., and Nebel, B. 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research* 14:253–302.
- Myers, K.; Tyson, M.; Wolverton, M.; Jarvis, P.; Lee, T.; and desJardins, M. 2002. PASSAT: A user-centric planning framework. In *Proceedings of the 3rd International NASA Workshop on Planning and Scheduling for Space*.
- Myers, K.; Jarvis, P.; Tyson, M.; and Wolverton, M. 2003. A mixed-initiative framework for robust plan sketching. In *ICAPS-03*.
- Nau, D.; Smith, S.; and Erol, K. 1998. Control strategies in HTN planning: Theory versus practice. In *IAAI-98*, 1127–1133.
- Pollack, M.; Joslin, D.; and Paolucci, M. 1997. Flaw selection strategies for partial-order planning. *Journal of Artificial Intelligence Research* 6:223–262.
- Smith, D., and Peot, M. 1993. Postponing threats in partial-order planning. In *AAAI-93*, 500–506.
- Tang, M., and Mali, D. 2003. Search control techniques for planning. In *15th IEEE International Conference on Tools with Artificial Intelligence (ICTAI-03)*, 168–175.
- Wolverton, M., and desJardins, M. 1998. Controlling communication in distributed planning using irrelevance reasoning. In *AAAI-98*, 868–874.