An Argument for a Hybrid HTN/Operator-Based Approach to Planning

Tara A. Estlin* and Steve A. Chien *' and Xuemei Wang ***

Jet I'repulsion Laboratory, California Institute of Technology†
4800 Oak Grove Drive, Pasadena, CA 91109-8099

Abstract. Work on generative planning systems has focused on two diverse approaches to plan construction. Hierarchical task network (HTN) planners build plans by successively refining high-level goals into lower-level activities. Operator-based planners employ means-end analysis to formulate plans consisting of Iow-level activities. While many have argued the universal dominance of a single approach, we present an alternative view: that in different situations either may be most appropriate. To support this view, we describe a number of advantages and disadvantages of these approaches in light of our experiences in developing two real-world, fielded planning systems.

1 Introduction

AI planning researchers have developed numerous approaches to the task of correct and efficient planning. Two main approaches to this task arc *operator-based* planners and *hierarchical task network* (HTN) planners. While considerable work has been done in analyzing and formalizing each of these approaches [Chapman 1987, Erol et al. 1994], and some work has been done in comparing them from a theoretical standpoint [Kambhampati 1995, Minton et al. 1 991], comparatively little effort has been devoted to comparing the two approaches in a more practical setting.

While both HTN and operator-based planners typically construct plans by searching in a plan-space, they differ considerably in how they express plan refinement operators. HTN planners generally specify plan mollifications in terms of flexible task reduction rules. Operator-based planners perform all reasoning at the lowest level of abstraction and provide a strict semantics for defining operator definitions. By virtue of their representation, HTN planners more naturally

' Current address: Department of Computer Science, University of Texas at Austin, Austin, TX, 78712-1188 estlin@cs.utexas.edu
*' Email address: steve.chien@jpl.nasa.gov
*** Cur-rent Address: Rockwell Science Center, 444 High St. Suite 400, Palo Alto, CA, 94.301, mei@rpal.rockwell.com
† This paper describes work performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

represent hierarchy and modularity. In contrast, operator-based plan refinements are more general since they can cover many more planning situations.

In this paper, we explain how a hybrid approach, which combines these two planning techniques, is an effective method for planning in real-world applications. In particular, we investigate the critical issue of planning representation. If domain knowledge can be naturally represented in a planning system then: (1) It will be easier to encode an initial knowledge base; (2) fewer encoding errors will occur, leading to a higher performance system; and (3) maintenance of the knowledge base will be considerably easier. Thus, an important measure for evaluating HTN and operator-based planning is how naturally each paradigm can represent key aspects of planning knowledge.

To evaluate representation abilities, we focus on four criteria: generality, hierarchy, flexibility, and efficiency. Generality describes the range of problem-solving situations that can be covered by a small amount of knowledge. Hierarchies allow common constraints, procedures, and patterns to be defined once yet used many times. Flexibility describes how easily a wide range of constraints can be accurately represented. Efficiency relates to how the representation influences the size of the planner's search space.

This paper describes a number of important representational issues that we have encountered in building two NASA planning systems [Chien et al. 1995]: Image Processing for Science Data Analysis (the MVP system) [Chien and Mortensen 1996] and Deep Space Network (DSN) Antenna Operations (the DPLAN system) [Chien et al. 1997]. The Multimission VICAR Planner (MVP) uses planning techniques to automatically generate image processing programs from user specified processing goals. MVP allows a user to specify a list of image processing requirements and then derives the required processing steps to achieve the input goals. Our second application concerns operating Radio Antennas. In this domain, the DPLAN planner is given a set of antenna tracking goals and equipment information. DPLAN then generates a list of antenna operation steps that will create a communications link with orbiting spacecraft.

Both of the planners described above employ a similar combination of HTN and operator-based planning techniques. Constructing and experimenting with these systems has helped us to closely examine many of the representation and efficiency trade-offs generated when using an integrated planning framework.

2 An Overview of HTN and Operator-based Planning

While we presume that the reader has a working knowledge of basic operator-based planning and HTN planning techniques, we briefly review the most salient differences of the two approaches.

An HTN planner [Erol et al. 1994] uses task reduction rules to decompose abstract goals into low level tasks. By defining certain reduction refinements, the user can direct the planner towards particular search paths. The user can also directly influence the planner by adding constraint information to a rule that would not strictly be derived from goal interaction analyses. HTN planners are

thus considered very flexible in representing domain information. Unfortunately, this flexibility can often lead to numerous overly -specific reduction rules that can be difficult to understand.

In contrast, an operator-based plannerm [Penberthy and Weld 1992, Carbonell et al. 1992, Weld 1994] reasons at a single level of abstraction - the lowest level. Actions are strictly defined in terms of preconditions and effects. Plans are produced through subgoaling and goal interaction analysis. All plan constraints are a direct consequence of goal achievements and precondition and effect analysis. This rigid representation is both a strength and a weakness. It is advantageous since it more explicitly directs the knowledge engineer in encoding a domain. Yet, it can also make certain aspects of a problem difficult to represent. For example, known ordering constraints can be difficult to encode if they cannot easily be represented in terms of preconditions and effects.

In an integrated HTN/operator framework, a planner can use multiple planning methods and reason about different types of planning goals. Both the MVP and DPLAN planners use a similar integration of HTN and operator-based planning methods. Domain information can be represented in either an HTN or operator format and both approaches can be used during planning to determine a problem solution. Domain information pertaining to these two techniques is kept separate; decompositional information is specified in decomposition rules, while items such as activity precondition and effects are kept in a separate schema list. This distinction is intended to allow a planner to apply a wider variety of planning, techniques and to formulate domain information in a flexible and usable representation. These planners can also easily use additional domain information for more efficient and flexible planning.

Two very related systems to MVP and DPLAN arc SIPE [Wilkins 1988] and O-Plan [l'ate et al. 1994], which both allow for the integration of HTN and operator-based planning[5]. However, O-Plait and SIPE do not retain as much of an explicit distinction between HTN and o~)etator-based planning techniques. Instead, typically plan formulation is primarily clone using decomposition operators (or networks). Operator-based features such as preconditions and effects are added to these structures when necessary. In contrast, we support an approach in which HTN planning and operator-based techniques can be used in conjunction or' as separate planning methods.

3 Representing Hierarchical and Modularity Information

Many of the obstacles in applying planning techniques to real-world problems can be characterized as representation difficulties. One advantage to employing an HTN planner is the ability to use abstract representation levels of domain objects and goals. Allowing abstract representations of these items enables us to represent domains in an object-oriented form, which is easier to write and reason about. This format also contributes to a more general domain knowledge base

[5] It is worth noting that these systems comprise 4 of the 5 applications recently described in [IEEE Expert 1996].

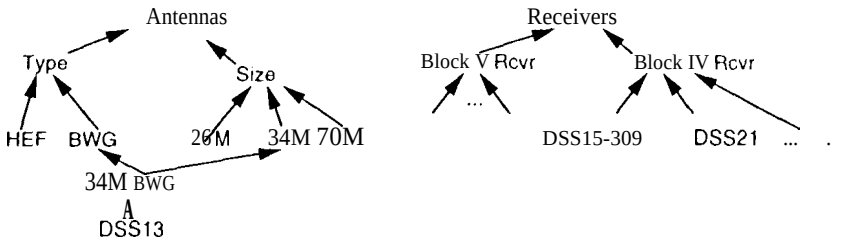

Fig. 1. Antenna and Receiver Hierarchies

that can be efficiently updated and maintained. Unlike operator-based planners, HTN planners provide direct support for this type of representation.[6]

3.1 **Object** and Goal Hierarchies

When using an HTN planner, different abstract levels O f domain objects and goals can be represented by constructing an object or goal hierarchy. More detailed information such as object instances is at one end of a hierarchy, while very general information such as broad object types is at the other end. In the DSN domain, different types of equipment are ofteu required for separate antenna activities. For example, many different types of antennas are currently represented in our domain. Our domain also includes several different types of receivers, which are used to receive data transmissions from orbit ing spacecraft. In Figure 1 we show partial equipment hierarchies for antennas and receivers.

The main advantage to this type of representation is that decomposition rules cau refer to either low- or high-level forms of a particular object or goal. In the DSN domain, a common antenna operation is performing a telemetry (or downlink) pass where information is transmitted from spacecraft to an antenna. A telemetry pass usually requires one of several types of receivers depending on the type of antenna being used. The maiu steps of the pass may be very similar for different antennas even though different receiver types are required. By using object and goal hierarchies we cau write just one telemetry decomposition rule to represent the general steps taken during this operation. For instance, in the telemetry rule shown in Figure 2, a general *pe7f07?11-receiver-co7 Ljgl17(ltio?l* goal is asserted as a new goal,

Information pertaining to specific equipment is contained in smaller, more specialized rules. For instance, specific receiver configuration steps can be added separately by decomposing the *perform-receiv er-configuration goal. The* rules listed in Figure 3 show two possible ways to break down this goal for either a Block-IV type receiver or a Block-V receiver. This format allows us to avoid writiug multiple versions of the main telemetry rule.

[6] For a discussion of these issues in the context of representing reactive planning knowledge see [Firby 1996].

```
(decomprule default-telemetry-track
  lhs
    (initialgoals ((track-goal spacecraft-track telemetry ?track-id)))
  rhs
    (newgoals ((g1 (perform-antenna-controller-configuration ?track-id))
               (g2 (perform-exciter-n-transmitter-configuration ?track-id))
               (g3 (perform-microwave-controller-configuration ?track-id))
               (g4 (~)crfort,,-receiver-co,,fiyurotio,, ?track-id))
               (g5 (perform-telemetry-configuration ?track-id))
               (g6 (move-antenna-to-point ?track-id))
               (g7 (perform-receiver-calibration ?track-id)))
    constraints ((before g1 g6)
                 (before g7 g3)
                 (before g4 g7))))
```

Fig. 2. Telemetry Decomposition Rule

```
(decomprule default-telemetry-track
  lhs
    (initialgoals ((perform-receiver-configuration ?track-id)))
    conditions (( CC N-equipment-assign ment ?track-id ?equip)
                ((isa ?equip BLOCK-IV-RECEIVER)))
  rhs
    (newgoals ((configure-block-iv-receiver ?track-id ?equip))))

(decomprule configure-receiver2
  lhs
    (initialgoals ((perform-receiver-configuration ?track-id)))
    conditions (( CCN-equipment-assignment ?track-id ?equip)
                ((isa ?equip BLOCK-V-RECEIVER)))
  rhs
    (newgoals ((configure-block-v-receiver ?track-id ?equip))))
```

Fig.3. Two Decomposition Rules for Receiver Configuration

By allowing object and goal hierarchies, we can construct domains in an object-oriented approach. Domain information is easily understood and updated since domain details arc kept separate from more general knowledge. For example, to understand the general steps of a telemetry operation, a user only has to view the main telemetry decomposition rule. If more low-level knowledge is desired, such as how to operate a particular piece of equipment, the user could search for rules that directly pertain to that equipment type. Knowledge maintenance is also more efficient. Most domain updates involve changes to only low-level steps. For instance, adding a new type of receiver to the domain, would not cause any rules that refer to more general receiver goals to be modified.

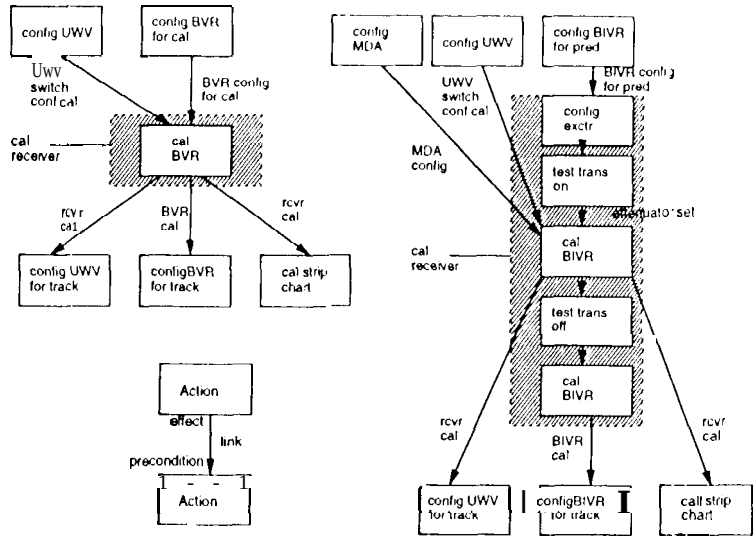

Fig. 4. VLBI Receiver Subplans

3.2 Modularity vs. **Specialized** Constraints

Unfortunately, a modular representation often makes it difficult to represent mote specialized inter-modular constraints. These types of constraints refer to information inside of several different decomposition rules and are usually only applicable in certain situations. Defining these constraints forces the addition of more specialized rules and often causes a hierarchical representation of rules to be infeasible,

For example, when performing receiver calibration in the DSN domain, it is sometimes necessary for high-level rules to refer to specific receiver calibration steps. When using a Block-IV receiver, VLBI (Very Long Baseline Interferometry) telemetry tracks directly impose high-level ordering constraints on specific receiver calibration steps, instead of on a more general *calib7ate-receiver goal.* Two different VLBI tracks are shown in Figure 4; the left uses a Block-V receiver and the right a Block-IV receiver. Low-level receiver calibration steps are shown in the shaded areas. In the Block-V case, receiver calibration is mapped onto a single general operator. However, in the Block-IV case it corresponds to five low-level steps which have constraints imposed on them by the telemetry rule. These constraints could he modified to refer to a more general goal (consisting of the shaded area), but then specialized constraint information would be lost. For instance, currently the *config-MDA* step and *config-exctr* step can be performed in parallel; however, if all ordering constraints are forced to refer to the entire shaded areas, such parallel execution would violate an ordering constraint,

One solution, which stays within the HTN framework, is to encode separate rules for tracks that require these inter-modular constraints. Unfortunately, this

solution results in less rule generality and increases the complexity of the domain definition. Another possibility is to represent the knowledge in a purely operator-based format. This option often provides a more compact representation of required constraint information, however, it has the disadvantages of (1) losing the representation hierarchy and (2) requiring more search.

A more satisfactory solution is to incorporate operator-based planning techniques with the hierarchical representation. Instead of directly adding these constraints to decomposition rules, we can implicitly represent them by adding preconditions and effects to low-level track steps. This approach permits intermodular ordering constraints to be separate from decomposition rules, thereby allowing rules to retain their modularity. Thus, in Figure 4, the link between *config-MDA* and the low-level *calibrate-Block-I V-receiver* step would be represented through preconditions and effects. The relevant ordering constraints would eventually be added through operator-based precondition achievement. The only drawback to this formulation is that acquiring constraints through goal achievement instead of specifying them directly in decomposition rules increases search. However, we feel this is an adequate tradeoff since it allows us to represent our domain information in a more useful and flexible format.

Point 1: Hierarchy and Modularity *HTN approaches have the advantage of easily supporting a hierarchical representation. Operator-based approaches have the advantage of generality, since they can cover many planning situations unconsidered by the knowledge engineer. Yet, they are usually less efficient. A hybrid HTN/operator- based approach allows an encoding that supports hierarchy and generality, without requiring an overly large search space.*

4 Encoding Implicit Constraints

Another advantage to using a hybrid planning system is the ability to encode implicit constraint information. These are constraints that may not be obvious when defining decomposition rules or operators, but are still necessary for correct planning. Consider the following example. When performing a telemetry pass in the DSN domain, a required step is to position the antenna to point at a specified set of coordinates (represented by the goal *move-antenna-to-point*). However, for many pre-calibration steps, which prepare the antenna for a transmission, it is necessary to have the antenna in a stow position where stray transmissions are directed at a harmless location. The antenna is not moved to point at the final coordinates until most pre-calibration steps have been executed. unfortunately, when defining the DSN domain, this constraint is often (accidentally) left out of many pre-calibration decomposition rules since it does not directly affect the success of pre-calibration activities.

One way to enforce this constraint is to explicitly acid ordering constraints to all telemetry decomposition rules that specify *move-antenna-to-point* be ordered *after* any activity that could cause the antenna to transmit. Unfortunately, such a constraint may have to be specified numerous times if there are multiple rules to which it applies. Another option is to use operator-basecl precondi-

tion/effect analysis. We could add a precondition of *not(antenna-at-point)* to any pre-calibration activities that could cause antenna transmission. This prevents the *move-antenna-to-point* step from being ordered before any pre-calibration activities that use the transmitter. Unfortunately, this option requires a number of extra preconditions to be added and could possibly induce more search.

A better solution is to utilize both HTN and operator-based techniques. First we can add a protection to the main telemetry decomposition rule that forbids stray transmissions during the entire pre-cal process. Then, using operator-based methods, we can require any pre-calibration transmission action to have a conditional effect which violates this requirement when the condition *antenna-at-point* is satisfied. This strategy requires pre-calibration actions that cause transmissions to be ordered before the action *move-antenna-to-point* is executed, and it causes the least amount of knowledge maintenance,

Point 2: Implicit Constraints *An HTN approach offers great flexibility* in specifying *arbitrary constraints, but may require* restating *constraints multiple times (when no appropriate hierarchy exists). Operator-based methods can also be used to represent these constraints however they often lead to* a *proliferation of operator preconditions. Hybrid methods offer the greatest flexibility in representing implicit constraints.*

5 Scripting vs. Declaring

Another notable difference between HTN arid operator based approaches is that the HTN approach allows the encoding of specific action sequences while an o~mrator-based approach often incurs significant search to construct this same sequence. Conversely, when operators can be combined in many different ways but still have interactions, an operator-based representation can be a more concise, natural method of encoding, these constraints. In varying domains, or portions of one domain, different aspects of these represent at ion t radeoffs are relevant. In order to demonstrate this tradeoff we performed an experiment where a knowledge engineer (KE) encoded a simplified portion of the MVP image processing domain [Chien and Mortensen **1996**].[7] This portion represented a subproblem called image navigation.[8] The KE developed three planning models, one in which only operator-based techniques were used, one where only HTN techniques were used, and one where both techniques were used.

All possible steps of the image navigation problem are shown in Figure 5. In the most basic case the process would involve setup steps A.1 and A.2, and automatic navigation steps B.4 and B.5. However, in some circumstances all asterisked steps would also be added. For example, if there is an initial tiepoint file, step A.3 might be added.

[7] The knowledge engineer had some knowledge of the image processing application and had no knowledge of this paper or research topic.

[8] This is perhaps the most complex subproblem in this image processing domain. It involves 8 top-level goals and 40 operators; a typical plan might range from 20-50 operators.

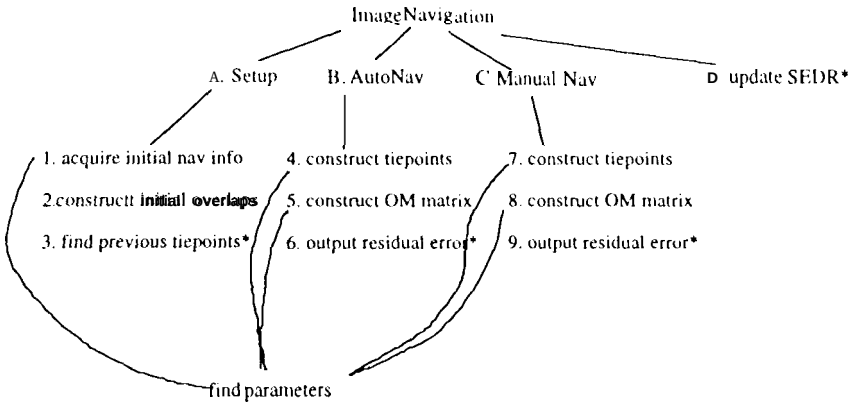

Fig. 5. MVP Navigation Process

Unfortunately, this is a very simplified navigation case. In most cases, the user would request a phase navigation process which would include more steps from both B and C. In these situations, manual navigation (C) would be performed to fine-tune the results of automatic navigation (B). To even more complicate matters, the exact specification of many steps depends on if other steps are being performed. For example, if residual error output is a requested goal, steps B.6 and C.9 must be executed. This requires that step B.5 and step C.8 have appropriate parameter settings to compute the residual output.

Furthermore, we have only listed the major component steps of navigating the image. There are also secondary steps that extract information from the image label. These secondary steps help appropriately select program parameters for each of the main steps listed in Figure 5. These extra details account for additional operators and steps in the plan not shown in Figure 5.

We compared the three knowledge bases constructed by the KE for this problem using the following measures: compactness of encoding, modularity (lack of repetition), and search efficiency. In Figure 6 wc summarize the number of HTN rules, number of operators, and search required for the most complex problems in each of the encodings.

Fig. 6. Knowledge Encoding Statistics

Encoding	# HTN Rules	Operators	Search
Operators	0	8	26
Rules	15	NA	5
Hybrid	5	8	18

Based on our results, the pure operator-based representation is inefficient from a search perspective. While only a small subset of the operator combinations will actually be used in solving problems, this type of framework requires that the

all operators be sufficiently accurate to rule out all other combinations. It is also difficult to debug the operators to ensure generation of only valid sequences. On writing a pure operator-based representation the KE said *"The operator KB was the most difficult to encode. One small change typically affected many operators and would require great re-testing. Because I had worked myself into a corner, I had to start from scratch a few times. For the final time, I realized that I needed to fully map the entire structure (including parameters) of paper."*

Representing this problem in a pure HTN framework is also difficult. Many complex combinations of dependencies and interrelations require numerous decomposition rules. Generally, there is one reduction rule for each basic sequence, and one rule for each combination of add-ons to the basic sequence. Unfortunately, this creates a proliferation of rules which are difficult to understand and maintain. The HTN encoding of this problem resulted in 4 rules to cover the automatic navigation process, 2 to cover the manual navigation process, and also a number of additional rules to address with previous tiepoint files (Step A.3). These rules account for the 15 rules required for the pure HTN representation.

In the combined HTN and operator-bawd framework it is possible to represent, different parts of the plan generation process using operator-based and/or HTN methods. Basic sequences can be easily represented using HTN rules. More complex additions to each basic sequence can be represented through operator-based constructs such as preconditions and conditional effects. Once the basic sequence has been determined through decomposition, goal-achievement is used to add additional constraints or dependencies. The complex navigation problem discussed above can now be represented as a separate script. For example, the two basic navigation phases, automatic navigation and manual navigation, can be represented in an HTN framework. However, slight modifications from the default framework (such as whether or uot to use an initial tiepoint file) can be linked in using operator-based planning techniques. This results in a reduced number of rules (compactness) and avoidance of redundancy in the KB. Avoiding redundancy is especially important since redundant portions of the KB must all be updated whenever one part is changed. This can lead to errors and increased maintenance costs.

Point 3: Scripting vs Declaring *An HTN framework is more search efficient than an operator-based one in cases where only a few sequences of operators are valid. An operator-based framework is representationally much cleaner, however, it requires a more general set of operators that can correctly manage many possible execution paths. In a hybrid framework, we can interleave the two planning processes (and representations) to produce an efficient planner that supports a compact, maintainable representation*

6 Other Representational Issues

6.1 Static Domain Information

One important issue in both operator-bascel and HTN-based planning is the ability to efficiently use static. state information to assist in pruning the search space.

Often, decomposition coalitions or operator preconditions can be considered static if they will remain unchanged throughout the planning process. These conditions can usually be evaluated immediately, which will help to initially prune the search space. Different planners are able to take advantage of this static information in varying degrees. In our integrated planning framework, static preconditions occuring in decomposition rules are labeled as such and only variable bindings which satisfy them are generated when considering applicable decomposition rules. Thus, codesignation commitment to satisfy static conditions occurs, but unnecessary commitment for other subgoals and variables is avoided. These static conditions are related to filter conditions [Pryor and Collins 1992] in that they are a specific type of filter condition restricting the applicability of the operator. However, precisely because static conditions cannot be changed by operators, they can be easily evaluated by a partial order planner and used in determining the applicability of a decomposition rule or operator.

6.2 Nominal Plan Generation

It is often desirable to predict (and control) the plans that are generated for nominal or near-nominal conditions. For example, when the problem goals or initial state change slightly, it is often desirable for the output plan to also change only slightly. This is a strong user requirement in both the image processing and DSN antenna operations applications. In operator-based planners, it is often difficult to encode such preferences. The planner would typically only be required to generate a correct plan. In contrast, since HTN planning techniques are closer to scripting, HTN planners offer good control over nominal or near-nominal plan generation. Hybrid HTN/operator planning frameworks can thus also offer control over nominal plan generation.

6.3 Replanning

A key requirement of many real-world planning systems is the ability to replan when plan goals or other conditions change. Replanning generally requires basic knowledge of why certain goals and actions are present in the plan. This requires a basic level of operator-based information and is mostly supported through techniques such as precondition and effect analysis. HTN approaches often encourage the omission of this information from the domain knowledge since it is not required for normal planning. In order to replan, hybrid techniques must still maintain any relevant precondition and effect information. Therefore, if replanning is necessary, much of the ease of an HTN encoding approach is lost because significant amount of operator-based information is still required.

6.4 Goal Modifiers

A relevant difference between operator-based and HTN planning is the number of goal modifiers that must be maintained. In operator-based planning, relevant

goal modifiers are listed as arguments to tile goal predicate. These modifiers then get propagated from goal to subgoal through operators. Thus, any parameters that are *possibly* relevant to a goal (and any of its subgoals) must be present as goal arguments. This procedure can result in long argument lists (often 10s of parameters), thereby increasing the difficulty of knowledge maintenance. In HTN planning, relevant modifiers are typically propagated top-down from abstract goals which expand into more specific activities. While this process still requires all possibly relevant parameters to be present, the expansions tend to result in short wide structures (e.g. an HTN rule expands a single goal into many goals). Thus, argument lengths quickly get shorter at lower levels Of abstraction. Unfortunately, a hybrid approach requires goal arguments to support both HTN and operator-based planning and hence offers no advantage over either.

7 Conclusion

This paper has described a number of issues relevant in representing planning knowledge in operator-based and HTN-based paradigms. We have described the main tradeoffs of using either HTN or operator-based specifications to represent domain knowledge. In particular, we discuss how these different methodologies impact the naturalness of the representation. HTN approaches are strong at modular and hierarchical representation, however operator-based approaches usually provide a more compact representation of constraints. Hybrid representations are best at managing the tradeoff between generality and efficiency. Hybrid approaches are also most flexible at encoding implicit constraints. HTN/hybrid approaches offer most control over nominal plan generation, but operator-based techniques offer the most support for replanning. HTN approaches most cleanly represent goal argument regressions. Based on these criteria we conclude that neither the operator-based approach nor the HTN approach dominates the other. Rather, in some cases the operator-based representation is more appropriate and in other cases the HTN representation is more appropriate. Thus, it seems most prudent to advocate usage of hybrid HTN/operator techniques.

References

[Carbonell et al. 1992] Carbonell, J.G.; Blythe, J.; Etzioni, 0.; Gil, Y.; Joseph, R.; Kahn, D.; Knoblock, C.; Minton, S.; Pérez, M. A.; Reilly, S.; Veloso, M.; and Wang, X. 1992. PRODIGY *4.0: The Manual and Tutorial.* Technical report, School of Computer Science, Carnegie Mellon University.

[Chapman 1987] D. Chapman, "Planning for Conjunctive Goals", 1987, *Artificial Intelligence* 32, 3.

[Chien and Mortensen 1996] S. A. Chien and H. B. Mortensen, " Automating Image Processing for Scientific Data Analysis of a Large Image Database," IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (8): pp. 854-859, August 1996.

[Chien et al. 1997] S. Chien, A. Govindjee, '1'. Estlin, X. Wang, A. Griesel, R. Hill Jr., Automated Generation of Tracking Plans for a Network of Communications Antennas, ['roe. 1997 IEEE Aerospace Conference, Aspen, CO, February, 1997.

[Chien et al. 1995] S. A. Chien, R. W. Hill Jr., X. Wang, T. Estlin, K . V . Fayyad, and H. B. Mortensen, "Why Real-world Planning is Difficult: A Tale of Two Applications)" I'roceeclingsoftl, el'bird European Workshop on Planning (EWSP95), Assisi, Italy, September 1995.

[Erol et al. 1994] K. Erol, J. Hendler, and D. Nau, "UMCP: A Sound and Complete Procedure for Hierarchical Task Network Planning," Proc. AIPS94, Chicago, IL, June 1994, pp. 249-254.

[Firby 1996] J. Firby, "Modularity Issues in Reactive Planning," ['roe. AIPS96, Edinburgh, UK, May 1996, pp. 78-85.

[IEEE Expert 1996] AI Planning Systems in the Real World, IEEE Expert, December 1996, pp. 4-12.

[Kambhampati 1995] Kambhampati, S., A Comparative Analysis of partial order planning and task reduction planning, SIGART Bulletin, Special Issue on Evaluating Plans, Planners, and Planning, Vol 6, No. 1, January 1995.

[Minton et al. 1991] Minton S., J. Bresina, and M. Drummond, "Commitment Strategies in Planning: A Comparative Analysis, " Proceedings AA AI-91.

[Penb erthy and Weld 1992] J. S. Penberthy and D. S. Weld, " UCPOP: A Sound Complete, Partial Order Planner for ADL," Proceedings of the Third International Conference on Knowledge Representation and Reasoning, October 1992.

[Pryor and Collins 1992] C. Collins and L. Pryor, "Achieving the functionality of filter conditions in a partial order planner," Proceedings AAA192, pp. 375-380.

[Tate et al. 1994] Tate, A., B. Drabble, and R. Kirby, "O-Plan2: An Open Architecture for Command Planning and Control," in *Intelligent Scheduling* (Eds. M. Fox and M. Zweben), Morgan Kaufmann, 1994.

[Weld 1994] An Introcduction to Least-commitment Planning, AI Magazine, 15(4) pp. 27-61, Winter 1994.

[Wilkins 1988] D. Wilkins. Practical *Planning: Extending the Classical A I Planning Paradigm*. Morgan Kaufmann, 1988.

This article was processed using the LaTeX macro package with LLNCS style