Metatheoretic Plan Summarization and Comparison

Karen L. Myers
SRI International
333 Ravenswood Ave.
Menlo Park, California, USA 94025
myers@ai.sri.com

Abstract

We describe a domain-independent framework for plan summarization and comparison that can help a human understand both key strategic elements of an individual plan and important differences among plans. Our approach is grounded in a *domain metatheory*, which specifies important semantic properties of tasks, instances and planning methods. The metatheory provides a semantic framework for guiding the choice and description of concepts used in summarizing and comparing plans, thus avoiding syntactic constructs whose meaning or import is unclear. We define three capabilities grounded in the metatheoretic approach: (a) summarization of an individual plan, (b) comparison of pairs of plans, and (c) analysis of a collection of plans. Application of these capabilities within a rich application domain shows their value in facilitating user understandability of complex plans.

Introduction

AI planning technology is being applied in increasingly more challenging application domains, resulting in the generation of plans with rich sophistication and complexity. In these domains, it is generally the case that a wide range of solutions is possible; part of the challenge for a human decision maker is to analyze the relative merits of various candidates before selecting a final option. The development of tools that can help users understand complex plans and tradeoffs among them thus presents an important technological challenge for the field.

In this paper, we describe an approach to hierarchical task network (HTN) plan summarization and comparison that is designed to help a human understand both key strategic elements of an individual plan and important differences among alternative plans. Plan summarization and comparison differ from *plan explanation*, which seeks to provide justifications for the causal structure of a plan, for example, why the plan contains a particular action, or how one action enables another. Explanation, while an important related technical capability, is not addressed in this paper.

Previous work on plan summarization and comparison has been grounded in methods that are tightly linked to either the syntactic characteristics of a plan's structure or the underlying reasoning processes used to generate it. Such approaches suffer from the problem that these structures and processes match the system's conceptualization of the domain rather than the user's. As such, their outputs have limited explanatory value.

Our approach to plan summarization and comparison is grounded in the use of a *domain metatheory*. The domain metatheory is an abstract characterization of a planning domain that specifies important semantic properties of its constituent tasks, instances and methods. The abstraction provides the means to describe and compare plans in high-level, semantically meaningful terms.

The concept of the domain metatheory was introduced originally to provide a language that would enable a user to *advise* a planning system, without requiring detailed knowledge of its internal workings [Myers 1996]. Advice, which describes high-level characteristics of desired solutions, is operationalized into structures and mechanisms that guide an automated planning system at runtime. Subsequently, the metatheory was also used as the basis for generating *qualitatively different* plans, by exploiting structure within the metatheory to direct a planning system toward solutions with distinct semantic traits [Myers & Lee 1999].

A key insight underlying the work reported here is that the metatheory can be used as the basis for identifying and communicating important descriptive information about a plan. In particular, the metatheory provides a semantic framework for guiding the choice of concepts used in summarizing and comparing plans. The resultant summaries and comparisons are thus grounded in semantically significant concepts rather than syntactic constructs whose meaning or import is unclear.

Within our metatheoretic framework, we define techniques for (a) summarization of an individual plan, (b) comparison of pairs of plans, and (c) analysis of a collection of plans. We focus in this paper on non-temporal aspects of a plan; temporal summarization and comparison are left for future work. Our techniques look for regularities or interesting exceptions relative to key aspects of the domain metatheory. For example, a metatheory *role* corresponds to an important actor or object within a plan. In comparing two plans, one pertinent dimension to consider is whether the plans fill key roles in different ways. Two plans may be similar in structure but one uses a cheap and abundant resource while the other relies on an expensive and more exotic resource.

Our approach embodies the spirit of *reconstructive explanation* [Wick and Thompson 1992], whereby an explanation is produced not by the problem solver's internal knowledge, but by a separate store of explanatory knowledge designed specifically with the user in mind. We believe that this style of approach is critical to ensuring that the results are of value to a user.

The plan summarization and comparison methods have been implemented within the PASSAT mixed-initiative planning framework [Myers et al. 2002]. To assess their effectiveness in facilitating user understandability of complex plans, we applied the methods to a test suite from an extensive special operations domain. This usage shows that our techniques can help a user understand subtle aspects of individual plans, important differences among plans, and the structure of the overall solution space.

Domain Metatheory

Our plan summarization and comparison work assumes an HTN planning framework, similar to that of [Erol et al. 1994]. An HTN domain theory consists of four basic types of element: *individuals* corresponding to real or abstract objects in the domain, *relations* that describe characteristics of the world, *tasks* to be achieved, and *methods* that describe available means for achieving tasks. We assume a type hierarchy for terms within the domain model. Thus, each individual has an associated type *Type(v)*, and there is a unique most-specific supertype *MinSupertype(V)* defined for any set of individuals *V*.

A *domain metatheory* defines semantic properties for domain theory elements that abstract from the syntactic details of the domain knowledge. Our metatheory for plan summarization and comparison extends and refines that introduced for the work on advisable planning [Myers 1996] in order to provide a somewhat richer and more structured framework. The main metatheoretic concepts that we use are *method features*, *task features*, and *roles*.

Method Features

A *method feature* designates a characteristic of a method that distinguishes it from other methods that could be applied to the same task. For example, among methods that could be applied to a transportation task, there may be an air-based method that is `fast` but `expensive` with a land-based alternative that is `slow` but `cheap`. Although the two methods are functionally equivalent in that they accomplish the same task, they differ significantly in their approaches. Method features provide the means to distinguish among such functionally equivalent alternatives by capturing these characteristics explicitly.

We model method features in terms of a *feature category* (e.g., `COST`) and a *feature value* (e.g., `cheap`). Feature values are drawn from a predefined *domain* for the feature category. Feature values may be quantitative or qualitative – for example, the feature category `COST` could have as its domain numeric cost values, or an enumerated set `{cheap medium expensive}`. We assume that feature domains are totally ordered.

We say that a *method feature f with value v occurs in plan P* iff there is some method *m* applied to a task *t* in *P* such that *m* has the feature *f* with value *v*. In general, a plan may have multiple occurrences of a given method feature that cut across methods used to accomplish a range of tasks. Different occurrences may have different values associated with them; duplication of values is also possible. The term *MFeatureInsts(f,P)* denotes the collection of values (including duplicates) for occurrences of method feature *f* in plan *P*.

The value of method features for plan summarization and comparison is that they provide the means to identify, abstract, and contrast important evaluational properties of different strategies, such as speed or cost. In particular, method features can be used as a kind of ‘quick and dirty’ proxy for deeper, more significant evaluations of a plan.

Task Features

Task features capture important semantic attributes of a task. As with method features, task features are modeled in terms of a feature category and feature value. In this paper, we focus on task features that designate *types* of activities, and restrict categories to the domain `[false true]`. For example, there may be several types of reconnaissance task: satellite reconnaissance, ground reconnaissance, and aircraft reconnaissance. Each of these tasks can be assigned the feature `RECON` with value `true`, thus providing a mechanism for abstracting over that set of tasks. (A similar sort of grouping could be achieved through the use of a taxonomy for tasks.) More generally, task features could be used to model a broader array of task properties, such as cost or risk.

We say that a plan *P has a task feature f* iff some task *t* in *P* has the feature *f* with value `true`. The term *TFeatures(P)* denotes the set of task features for *P*.

Roles

A *role* describes a capacity in which an individual is used within a method or task; it maps to a method or task variable. For instance, a method for transporting materials may contain variables `location.1` and `location.2`, with the former corresponding to the `START` role and the latter the `DESTINATION` role for the move.

Roles provide a semantic basis for describing the use of individuals within methods and tasks that abstracts from the details of specific variable names. Roles also provide the means to reference a collection of semantically linked variables that span different methods and tasks (i.e., `START` roles may occur in multiple methods and tasks).

We say that a *role r with fill v occurs in plan P* iff either:

- there is some task $t(a_1, \ldots a_n)$ in *P* such that *t* has the declared role *r* for its i^{th} argument, and $a_i = v$ [Task Role], or
- some method with role *r* declared for local variable x_i is applied to a task $t(a_1, \ldots a_n)$ in *P*, and x_i is bound to *v* [Method Role]

The term *Roles(P)* denotes the set of roles that occur in plan *P*, while *RoleFills(r,P)* denotes the values (including duplicates) that occur as fills for role *r* in plan *P*.

Experimental Framework

We evaluated the effectiveness of our plan summarization and comparison techniques on a suite of nine test plans drawn from a *special operations forces* (SOF) domain. The SOF domain constitutes a sizable and rich test environment for evaluating our work: it contains 65 predicates modeling relevant world properties, more than 100 tasks, and more than 50 methods spanning five abstraction layers.

The SOF domain was created during an earlier project on mixed-initiative planning technology. The original domain included a limited metatheory designed to showcase advisability within the PASSAT system [Myers et al. 2002]. For the work on plan summarization and comparison, the domain was extended to include a fairly comprehensive metatheory with 13 method features, 12 task features, and more than 75 roles. The task features (see Figure 3) use the domain `[false true]`; the method features (see Figure 4) are modeled qualitatively using the domain `[low medium high]`.

The test plans address the objective of extracting a set of hostages held by a guerilla team in an urban environment. More specifically, this task requires rescuing a set of hostages being kept at Mogadishu-Town-Hall using forces based at Riyadh Airport, and then evacuating the hostages to a safe haven at Riyadh Stadium.

The SOF domain includes a number of methods that reflect different strategies for rescuing the hostages. Variations among solutions result from three sources. The first is whether the plan contains certain types of strategic and tactical activities; depending on a given situation, the commander may decide to include such activities within the plan. For example, while it is not necessary to create diversions to distract the guerillas, doing so may be desirable in some circumstances. The second relates to the selection of resources to be used. In some cases, for example, it may be appropriate to use satellites to gather intelligence information while in others it may be preferable to rely on ground forces. The third relates to decisions about key parameters within a plan, such as where to establish a forward base.

Figure 1 summarizes the nine test plans used in our evaluation. These plans were created by the developer of the SOF domain through a combination of manual and semiautomated methods within PASSAT. This individual was not involved with the research on plan summarization and comparison described in this paper. As such, the plans provide an impartial test suite for evaluating our work.

The plan developer was asked to create a core set of plans reflecting a representative set of strategic alternatives that a SOF commander might consider. In addition, he was asked to create variants of the core plans by making a few strategic changes that might correspond to handling contingencies in different ways. Given that variants of this type are made frequently in practice, we were interested in determining how well our plan comparison techniques could identify differences among them.

Plan ID	Description	#Tasks
tiny-a	Very simple plan without security or support	7
tiny-b	Variant on *tiny-a* that uses a different type of rescue force	7
small-a	Basic plan that includes reconnaissance and combat search and rescue	20
small-b	Variant on *small-a* that uses the same high-level strategy but differs in the lower-level realization of parts of it	18
medium-a	Broadly similar to the small plans but involves refueling	38
medium-b	Broadly similar to the small plans but with suppression of enemy air defenses (SEAD) activities	50
large-a	Extensive plan with significant reconnaissance and support, and a diversion from the main assault	125
large-b	Variant on *large-a* that provides increased fire support and SEAD	135
large-c	Variant on *large-a* with a different style of diversion	136

Figure 1. Summary of Test Plans

Although we focus on the SOF domain in this paper, our plan summarization and comparison methods are domain independent and broadly applicable. The methods are well suited to planning domains for which (a) problems have numerous solutions, (b) the solutions are complex in scope, and (c) solutions vary significantly in their content. Such domains embed interesting *design* problems, for which planners must make important decisions that trade off factors such as quality, risk, and cost. Many military planning problems have this flavor; other suitable domains include process planning, exploratory robot planning [Bresina et al. 2005], and business workflow planning. Our techniques are not well suited to 'puzzle' problems (e.g., blocks world), where the challenge is to find one of a small number of solutions with similar logic and structure.

Plan Summarization

The roles and features of the metatheory provide a semantic basis for summarizing key properties of a plan. In particular, a description of how a plan fills its roles and the features that it possesses can provide valuable insight into the structure, strengths, and weaknesses of a plan.

Task Features

Task features provide a succinct summary of key activities within a plan. Such a summary can inform the user that a given plan does or does not contain activities such as refueling or fire support. The task features for a given plan can be computed easily by traversing the plan and collecting features for each task encountered.

Method Features

Method features provide a different perspective on a plan, as they designate plan characteristics of an evaluational nature (e.g., cost, speed). A given method feature can appear multiple times in a plan, with different occurrences yielding different values. This variation reflects the fact that, for example, a given plan may use an inexpensive reconnaissance operation but an expensive rescue strategy.

To enable plan-level summarization of the property represented by a method feature, we introduce the concept of the *method feature value* for a feature f and plan P. This value is derived from *MFeatureInsts(f,P)* – the collection of feature values associated with occurrences of f in P. For a quantitative method feature f, the method feature value for P is the sum of the values in *MFeatureInsts(f,P)*; for a qualitative method feature, the value is the *qualitative average* (defined below) over these values.

Definition 1 [Method Feature Value for a Plan] The *method feature value* for feature f and plan P, denoted by *MFeatureValue(f,P)*, is defined to be *QualAvg (MFeatureInsts(f,P))* for qualitative features, and *Sum(MFeatureInsts(f,P))* for quantitative features.

Qualitative averaging can be done in various ways. We require for each qualitative feature f a surjective, order-preserving mapping θ_f from a designated interval of the reals *Interval*(f) to the domain of f: θ_f: *Interval*(f) → *Domain(f)*. Variation in the 'closeness' of values in *Domain(f)* can be achieved by appropriate definitions of θ_f.[1] With this mapping, we define the average of a set V of qualitative feature values as follows:[2]

$$QualAvg(V) = \theta_f\left(\frac{1}{|V|} \times \sum_{v \in V} \theta_f^{-1}(v)\right)$$

Roles

A description of how roles are filled within a plan can provide a concise summary of what resources are used and how, as well as key choices for a plan (e.g., the location for a forward base). Furthermore, it is possible to search for patterns in the filling of roles. So, for example, it may be useful to know that all transport of troops is through helicopters of a particular type. We refer to such patterns as *uniformities* for role fills and define two categories of uniformity, oriented around *values* and *types*.

[1] Every method feature in the SOF metatheory has the domain [`low medium high`], the interval [0,1], and the mapping θ: [0,1] → [`low medium high`] with θ^{-1} distributed linearly across [0,1] (i.e., `low` maps to 0, `medium` to 0.5 and `high` to 1).

[2] This definition of qualitative averaging does not account for the relative importance of different feature instances. For example, consider a travel plan in which the qualitative method feature `COST` has five occurrences: one with value `high` for the purchase of a first-class airline ticket and four with values `low` for the purchase of bus tickets for travel to/from the source and destination airports. Using the definition of *QualAvg* above, the method feature value for `COST` in this plan would be `low` when in reality the overall cost of the trip would be high relative to other options. This problem can be solved through the use of qualitative features at differing levels of granularity; in the travel example, there could be one feature for significant expenses and another for incidental expenses.

Value uniformity arises when there is more than one occurrence of a role within a plan (i.e., |*RoleFills(r,P)*| > 1) but all are filled by the same value.

Definition 2 [Value Uniformity in Role Fills] A plan P *uniformly fills a role r with value c* iff |*RoleFills(r,P)*| > 1 and $v \in$ *RoleFills(r,P)* implies that $v = c$.

Type uniformity builds on the declared type *Type(r)* for a role r, which indicates that fills for r must be of that type. Type uniformity is interesting when some proper subtype of *Type(r)* generalizes all fills for a given role, thus showing a potentially significant specialization of the type. For example, it can be useful to note that only satellites are used for reconnaissance within a plan, although other asset types (e.g., ground forces) are possible.

Definition 3 [Type Uniformity in Role Fills] A plan P *uniformly fills a role r with a type T* iff |*RoleFills(r,P)*| > 1, T is a proper subtype of *Type(r)*, and every fill value $v \in$ *RoleFills(r,P)* is of type T.

Value and type uniformities for roles constitute generic, domain-independent mechanisms for generalizing a collection of role fills. For a given domain, it may be appropriate to introduce additional domain-specific generalization mechanisms. For example, in domains where locations play a significant role, it might be useful to generalize based on geographic proximity, or co-location within some designated geographic area (e.g., all air assets are pulled from bases in the same region).

Sample Plan Summary

To illustrate the value of metatheory-based plan summarization, consider the test plan *medium-b* shown in Figure 2. The figure presents a task decomposition view of the plan that highlights its hierarchical structure; for simplicity, temporal sequencing information has been omitted. As can be seen, the plan is sufficiently complex that its key strategic elements, merits and weaknesses are not readily apparent. Rather, tools of some sort are needed to facilitate understanding of the plan.

Figures 3–6 present the summarization for this plan produced by our metatheoretic techniques. Figure 3 summarizes the task features; Figure 4 summarizes the method features and their normalized values; Figure 5 summarizes key role fills; Figure 6 summarizes role uniformities.

The summary of task features in Figure 3 makes it easy to identify the key strategic elements of the plan. The features `RESCUE-AND-RECOVER` and `RESCUE` derive from the fact that the plan constitutes a rescue-and-recover operation; these features are common to every plan in the test suite. At a lower level, we can see that this solution includes components for combat search and rescue support (`CSAR-SUPPORT`), fire support (`FIRE-SUPPORT`), reconnaissance (`RECON`), and suppression of enemy air defenses (`SEAD`). These more specialized components are notable, as not every solution contains them.

```
* (Rescue-Hostage Mogadishu-Town-Hall Riyadh-Airport Riyadh-Stadium)
  * (Rescue-And-Recover Riyadh-Airport Mogadishu-Town-Hall Riyadh-Stadium)
    * (Recon Mogadishu-Town-Hall)
      * (Infiltrate Green-Oda-2 Ankara-Airport Mogadishu-Town-Hall)
        * (Produce-Landing-Plan Mh-60-G-Pave-Hawk-2)
        * (Produce-Air-Movement-Plan Mh-60-G-Pave-Hawk-2)
        * (Produce-Loading-Plan Green-Oda-2)
        * (Produce-Aircraft-Bump-Plan Green-Oda-2)
        * (Load Green-Oda-2 Mh-60-G-Pave-Hawk-2)
        * (Fly Mh-60-G-Pave-Hawk-2 Ankara-Airport Mogadishu-Stadium)
        * (Drop Green-Oda-2 Mh-60-G-Pave-Hawk-2 Mogadishu-Town-Hall)
        * (Depart Mh-60-G-Pave-Hawk-2 Mogadishu-Stadium)
      * (Establish-Observation-Post Green-Oda-2 Mogadishu-Town-Hall)
      * (Exfiltrate Green-Oda-2 Mogadishu-Town-Hall Mogadishu-Building4)
        * (Fly Uh-60a-2 Mogadishu-Town-Hall Mogadishu-Building3)
        * (Load Green-Oda-2 Uh-60a-2)
        * (Depart Uh-60a-2 Mogadishu-Building3)
      * (Provide-Fire-Support Mogadishu-Town-Hall)
        * (Take-Off Ch-53e-Super-Stallion-1 Addis-Ababa-Airport)
        * (Fly Ch-53e-Super-Stallion-1 Addis-Ababa-Airport Mogadishu-Town-Hall)
        * (Place-On-Station-Fire-Support Ch-53e-Super-Stallion-1 Mogadishu-Town-Hall)
        * (Fly Ch-53e-Super-Stallion-1 Mogadishu-Town-Hall Addis-Ababa-Airport)
        * (Land-At Ch-53e-Super-Stallion-1 Addis-Ababa-Airport)
      * (Provide-Csar-Coverage Csar-Team-2 Mogadishu-Town-Hall)
        * (Prepare Csar-C1-A)
        * (Take-Off Csar-C1-A Baidoa-Stadium)
        * (Fly Csar-C1-A Baidoa-Stadium Mogadishu-Town-Hall)
        * (On-Station Csar-C1-A Mogadishu-Town-Hall)
        * (Provide-Fire-Support Mogadishu-Town-Hall)
          * (Take-Off Ah-100-1 Balikesir-Stadium)
          * (Fly Ah-100-1 Balikesir-Stadium Mogadishu-Town-Hall)
          * (Place-On-Station-Fire-Support Ah-100-1 Mogadishu-Town-Hall)
          * (Fly Ah-100-1 Mogadishu-Town-Hall Balikesir-Stadium)
          * (Land-At Ah-100-1 Balikesir-Stadium)
       * (Provide-Sead Sead-1 Ad-Dammam-Stadium Mogadishu-Town-Hall)
          * (Prepare Sead-1)
          * (Take-Off Sead-1 Ad-Dammam-Stadium)
          * (Fly-To Sead-1 Mogadishu-Town-Hall)
    * (Infiltrate Orange-Oda-1 Riyadh-Airport Mogadishu-Town-Hall)
      * (Fly-Commercial Aa7864 Orange-Oda-1 Riyadh-Airport Mogadishu-Town-Hall)
    * (Storm Orange-Oda-1 Mogadishu-Town-Hall)
    * (Exfiltrate Orange-Oda-1 Mogadishu-Town-Hall Riyadh-Stadium)
      * (Fly-Commercial Aa201 Orange-Oda-1 Mogadishu-Town-Hall Riyadh-Stadium)
    * (Provide-Fire-Support Mogadishu-Town-Hall)
      * (Station Yorktown Mogadishu-Town-Hall)
    * (Provide-Csar-Coverage Csar-Team-2 Mogadishu-Town-Hall)
      * (Prepare Uh-60l-1)
      * (Take-Off Uh-60l-1 Bihac-Stadium)
      * (Fly Uh-60l-1 Bihac-Stadium Mogadishu-Town-Hall)
      * (On-Station Uh-60l-1 Mogadishu-Town-Hall)
```

Figure 2. Task Decomposition View of Plan medium-b

The method features in Figure 4 summarize key evaluational qualities of the plan. Desirable qualities include the fact that the expected quality of information underlying the plan is high, while expected collateral damage is low. On the negative side, there is high vulnerability to ground fire.

More than 30 roles occur in the plan *medium-b*, some with multiple fills. Typically, a user would not choose to view all roles and their fills at once. Rather, at a given point in time he would be interested in focusing on certain aspects of the plan. So, for example, a user interested in understanding the high-level strategy of a plan may concentrate on a subset of roles related to key strategic decisions, while a user interested in asset usage may concentrate on roles related to resource utilization.

Figure 5 displays the role fills related to force usage for the plan *medium-b*. For fill values that occur more than once for a given role, the number of occurrences is noted in parentheses. This summary makes it easy to see that only Green and Orange teams are used in the plan; both are used in assault roles while the Green team is also used in a reconnaissance capacity as an observation force. The color in a force name is significant as colors denote units with specific skills and capabilities; space limitations preclude detailed descriptions of these color-related characteristics.

Figure 6 summarizes value-based and type-based role uniformities for the plan *medium-b*. For value-based uniformity, the summary indicates the role, fill value, andnumber of occurrences. For type-based uniformity, the summary indicates the role, its type, the types of the fill values, and the most specific type that generalizes the fill values. The information on type-based uniformity is particularly useful here as it highlights the exclusive use of air assets for key functional roles within the plan.

Task Features Present	Task Features Absent
CSAR-SUPPORT FIRE-SUPPORT RECON RESCUE RESCUE-AND-RECOVER SEAD SUPPORT	DIVERSION EVACUATION PARACHUTE REFUELING SECURITY

Figure 3. Task Feature Summary for Plan *medium-b*

Method Feature	Plan Value
BLUE-CASUALTY-RISK	medium
COLLATERAL-DAMAGE	low
COORDINATION-COMPLEXITY	medium
COVERTNESS	medium
DURABILITY	low
FORCE-FATIGUE	medium
FORCE-FOOTPRINT	medium
FORCE-INTEGRITY	medium
INFORMATION-QUALITY	high
LANDING-ZONE-PREP	low
ROBUSTNESS	medium
SPEED	medium
VULNERABILITY-GROUND-FIRE	high

Figure 4. Method Features for Plan *medium-b*

Role	Fill Values
ASSAULT-FORCE	green-oda-2 orange-oda-1
FORCE	orange-oda-1 (2) green-oda-2 (2)
OBSERVATION-FORCE	green-oda-2

Figure 5. Force Usage Roles in medium-b

Value-based Role Uniformity

Role	Value	Count
CSAR-LOCATION	mogadishu-town-hall	2
FIRE-SUPPORT-LOCATION	mogadishu-town-hall	5
FORWARD-POINT	riyadh-airport	3
INFIL-DESTINATION	mogadishu-town-hall	2
SEAD-AIRCRAFT	sead-1	2

Type-based Role Uniformity

Role	Role Type	Fill Types	Min. SuperType
EXFIL-ASSET	Asset	Commercial-flight Helicopter	Air-asset
INFIL-ASSET	Asset	Commercial-flight Helicopter	Air-asset
TRANSPORT-ASSET	Asset	Sead-aircraft Helicopter	Air-asset

Figure 6. Role Uniformities in Plan *medium-b*

Plan Comparison

We define two techniques for comparing plans: *feature differencing* and *role differencing*. They are useful for both identifying subtle variations in similar plans and understanding larger differences in more varied plans.

Feature Differencing

As noted above, features correspond to high-level semantic characteristics of tasks (for task features) and strategic or evaluational qualities (for method features).

Task feature differencing, which involves a comparison of task features within two plans, provides a snapshot of how the two plans differ in their key task types. This capability can enable a user to see easily that, for example, one plan contains reconnaissance capabilities while another does not.

Method feature differencing compares the normalized method feature values for two different plans in order to identify significant variations. This form of differencing makes it easy to see, for example, that one plan trades risk for increased complexity relative to another plan.

Role Differencing

Role differencing looks at variabilities in how two plans fill their roles. This type of comparison can shed insight on key differences in strategic decisions (e.g., *Where will the hostages assemble?*) and resource usage (e.g., *What types of reconnaissance asset are used?*).

Figure 7 presents a categorization of the ways in which the fill values for a given role in two plans can differ. There, V_1 and V_2 designate sets of fill values for a role from which duplicates have been removed. It is assumed that $V_1 \neq V_2$ and that both V_1 and V_2 are nonempty. The first three entries cover situations where V_1 and V_2 are disjoint; the last two cover situations where V_1 and V_2 overlap.

The category *different single valued,* although just a special case of *disjoint types*, is useful for identifying differences in strategic decisions for a plan. For example, for the role `ASSAULT-FORCE`, the plan *tiny-a* uses `orange-oda-2` while the plan *tiny-b* uses `green-oda-1`. This difference is important, as noted above, because orange and green forces have significantly different core capabilities. The category *different single valued* is especially useful when the role appears exactly once within each of the two plans being compared; such a role often designates some critical parameter choice.

The category *disjoint types* requires both that the most specific supertype of the role-fill values in the two plans differ, and that neither be a subtype of the other. For example, the plan *small-a* uses only helicopters of type `CSAR-HELICOPTER-CLASS-1` for combat search and rescue while the plan *large-b* uses helicopters of type `CSAR-HELICOPTER-CLASS-2`. The category *disjoint multivalued* defines an even weaker condition, requiring only that the fill values for the two plans differ.

For overlapping values, the strongest condition is *restricted subtype*, which indicates that the most specific supertype of one collection of values is a subtype of the most specific supertype of the other collection. For example, the plan *large-b* uses only assault forces of type `SOF-UNIT` while the plan *large-c* uses a more general set of forces (of type `FORCE-COMPOSITION`); in contrast, the plan *large-b* uses a range of watercraft to fill the role `WATER-ASSET` while the plan *large-c* uses only values of type `BOAT`. *Restricted subset* weakens the *restricted subtype* condition to require only that one collection of values be a subset of the other.

Disjoint: $V_1 \cap V_2 = \{\}$

Different single valued: $V_1 \cap V_2 = \{\} \wedge |V_1|=|V_2|=1$

Disjoint types: $\mathrm{MinSupertype}(V_1) \neq \mathrm{MinSupertype}(V_2) \wedge \mathrm{MinSupertype}(V_1) \not\subset \mathrm{MinSupertype}(V_2) \wedge \mathrm{MinSupertype}(V_2) \not\subset \mathrm{MinSupertype}(V_1)$

Disjoint multivalued: $V_1 \cap V_2 = \{\} \wedge (|V_1|>1 \vee |V_2|>1)$

Overlapping: $V_1 \cap V_2 \neq \{\}$

Restricted subtype: $\mathrm{MinSupertype}(V_1) \subset \mathrm{MinSupertype}(V_2) \vee \mathrm{MinSupertype}(V_2) \subset \mathrm{MinSupertype}(V_1)$

Restricted subset: $V_1 \subset V_2$

Figure 7. Categories of Role-fill Differences

Role differencing can provide insights into fundamental differences between plans, as illustrated in the next section. However, there are limitations to its usefulness.

First, the significance of role differences may be difficult to gauge in isolation. So, while the decision to use force `Green-ODA-1` rather than `Orange-ODA-2` to fill the `ASSAULT-FORCE` role is significant, as those two units have markedly different capabilities, the difference between the forces `Green-ODA-1` and `Green-ODA-2` is insignificant as they have the same fundamental capabilities. This problem can be addressed by introducing a notion of 'semantic distance' between individuals to help identify differences that are significant.

Second, the utility of role differencing can decrease as plan size grows due to increased numbers of occurrences of a role that are not closely related. (For example, it is possible to create larger SOF plans by introducing multiple assault prongs involving forces inserted at different drop locations; doing so leads to duplication of roles used in very different contexts.) Thus, while unrestricted role differencing can be useful in small- to medium-sized plans, larger plans would benefit from a scheme to contextualize role fills to portions of the plan. This point is addressed further in the Discussion section below.

Sample Plan Comparison

Figure 8 displays the results of applying our metatheoretic plan comparison techniques to the test plans *medium-a* and *medium-b*. In looking at the results of task feature differencing, two fundamental differences emerge: *medium-a* contains refueling activities and *medium-b* does not, while *medium-b* contains SEAD (suppression of enemy air defense) activities and *medium-a* does not.

For method feature differencing, there is some variation among expected values for key evaluation criteria. Given

Task Feature Differencing:
Task Features in *medium-a* but not in *medium-b*: `REFUELING`
Task Features in *medium-b* but not in *medium-a*: `SEAD`

Method Feature Differencing:

MethMethod Feature	*medium-a*	*medium-b*
DURABILITY	medium	low
FORCE-FATIGUE	high	medium
FORCE-INTEGRITY	high	medium
LANDING-ZONE-PREP	medium	low

Role Differencing:

Different Single Valued

Role	*medium-a*	*medium-b*
RECON-ASSET	satellite-1	green-oda-2

Disjoint Multivalued

Role	Values for *medium-a*	Values for *medium-b*
ASSET	csar-c2-b tanker-1	uh-60l-1 yorktown sead-1 csar-c1-a
EXFIL-ASSET	uh-60l-1	aa201 uh-60a-2
FIRE-SUPPORT-ASSET	av-8b-harrier-ii-a	yorktown ah-100-1 ch-53e-super-stallion-1
INFIL-ASSET	uh-60l-2	aa7864 mh-60-g-pave-hawk-2
TRANSPORT-ASSET	tanker-1 av-8b-harrier-ii-a uh-60l-1 uh-60l-2	sead-1 uh-60a-2 mh-60-g-pave-hawk-2

Restricted Subtype

Role	Type for *medium-a*	Type for *medium-b*
ASSAULT-FORCE	ORANGE-UNIT	SOF-UNIT
INFIL-POINT	BUILDING	POINT-LOCATION
INFIL-TEAM	ORANGE-UNIT	SOF-UNIT
LANDING-LOCATION	AIRPORT	POINT-LOCATION

Restricted Subset

Role	Values for *medium-a*	Values for *medium-b*
EXFIL-POINT	mogadishu-town-hall	mogadishu-town-hall mogadishu-building4
INFIL-START	riyadh-airport	riyadh-airport ankara-airport

Figure 8. Comparison of *medium-a* and *medium-b*

the use of a fairly coarse-grained set of qualitative values for method feature domains in the SOF metatheory, the scope for variability is limited. A finer-grained set of values would enable more precise comparisons.

Role differencing highlights some interesting variations in resource use for the two plans. Both plans include reconnaissance operations, but *medium-a* employs a satellite (`satellite-1`) while *medium-b* employs a ground force (`green-oda-2`) as the asset used to perform the reconnaissance (see the table *Different Single Valued*). This distinction is important because the nature and quality of the intelligence that can be obtained from these two assets differs markedly. Different types of infiltration, exfiltration, fire support and transport assets are used, each with their individual strengths and weaknesses (see the table *Disjoint Multivalued*). The tables *Restricted Subtype* and *Restricted Subset* show that plan *medium-a* is less diverse than plan *medium-b*, since it uses more restricted sets of entities to fill a number of key roles (i.e., `ORANGE-UNIT` is a subtype of `SOF-UNIT`, `BUILDING` and `AIRPORT` are subtypes of `POINT-LOCATION`).

Overall, a user looking at the comparison in Figure 8 could grasp quickly the fundamental differences in strategy and resource usage between the two plans. Direct examination of the plans shows additional differences in terms of unimportant low-level activities; the metatheoretic comparison hides these nonessential differences.

Plan Space Analysis

We define two capabilities grounded in the domain metatheory for reasoning about a collection of plans: *identifying unique characteristics of a plan*, and *identifying maximally different plans*.

Identifying Unique Characteristics of a Plan

The metatheoretic differencing capabilities defined in the previous section can be used to identify three useful distinguishing characteristics of a plan *P* relative to a set *S* of candidate solutions.

1. *Unique task features:*
   - *P* has a task feature not found in any other *P'* in *S*
   - *P* lacks a task feature found in all *P* in *S*
2. *Unique normalized method features: P* has a normalized method feature value that differs from the value for all other solutions in *S.* This situation is especially interesting when all other plans share a common value for that method feature; in that case, the method feature for plan *P* is called *exceptional*.
3. *Differing role fills:* There is a role common to all plans for which some fill value in *P* does not occur as a fill value in other solutions in *S*.

Figure 9 summarizes the unique task features and normalized method features for our suite of test plans; they occurred in the plans *small-b* and *medium-a*. (We have not yet implemented the ability to look for differing role fills.)

The plan *small-b* differs from all others in the test suite on the normalized value for the method feature `BLUE-CASUALTY-RISK`. In particular, its value for that feature is `low` while the other plans have the value `medium`.

The plan *medium-a* has several unique characteristics relative to the other plans in the test suite. First, it is the only plan with the task feature `REFUELING`; hence, no other plans in the test suite include refueling operations. Second, while the plan *medium-a* has the normalized value `medium` for the method feature `LANDING-ZONE-PREP`, all other plans have the value `low`. Finally, the plan *medium-a* differs from the other plans in the values for method features `DURABILITY` and `FORCE-FATIGUE`; in

Plan: *small-b*
Has Exceptional Method Feature Values:
`BLUE-CASUALTY-RISK: low;` all others `medium`

Plan: *medium-a*
Has Unique Task Features: `REFUELING`
Has Exceptional Method Feature Values:
`LANDING-ZONE-PREP: medium;` all others `low`
Has Unique Method Feature Values:
`DURABILITY: medium`
`FORCE-FATIGUE: high`

Figure 9. Unique Features in the Test Suite

those cases, however, there is no common value for the remaining plans in the test suite.

Maximally Different Plans

For many applications, a human planner will want to explore a range of plans that embody qualitatively different solutions [Tate et al. 1998; Myers & Lee 1999]. Exploration of this type can be useful both in terms of helping the user understand fundamental tradeoffs that are inherent to the domain, and identifying 'out of the box' solutions that he may not normally consider.

Our metatheoretic differencing techniques can be used to identify plans that are semantically far apart from each other, and hence are likely to have significant qualitative differences. To that end, we define a concept of *distance* between plans that builds on the concepts of *task feature*, *method feature*, and *role distance* between plans.

Task Feature Plan Distance

Task feature distance is a normalized form of Hamming distance for the task features within the plans. In particular, it is defined to be the ratio of the number of task features that appear in one but not both plans to the number of features that appear in either plan. Below, we make use of the following definition:

$$HammingDist(V1, V2) = \frac{|V1 - V2| + |V2 - V1|}{|V1 \bigcup V2|}$$

Definition 4 [Task Feature Plan Distance] The *task feature distance* between plans P_1 and P_2 is defined to be *TFeatureDist*(P_1,P_2) = *HammingDist*(*TFeatures*(P_1), *TFeatures*(P_2)).

Method Feature Plan Distance

Method feature distance for a pair of plans is defined to be the average distance between the values of those features that are common to both plans, normalized with respect to the range of possible values for the features. Let *MethodFeatures(P)* denote the collection of method features that occur in plan *P*, and $FDist(f,P_1,P_2)$ the normalized distance (defined below) between values for method feature *f* in plans P_1 and P_2.

Definition 5 [Method Feature Plan Distance] The *method feature distance* between plans P_1 and P_2, denoted by $MFeatureDist(P_1, P_2)$, is defined as follows, where $F^{\cap}$=*MFeatures*(P_1) ∩ *MFeatures*(P_2).

$$MFeatureDist(P_1, P_2) = \frac{1}{|F^{\cap}|} \times \sum_{f \in F^{\cap}} FDist(f, P_1, P_2)$$

For quantitative feature values, $FDist(f,P_1, P_2)$ is defined as

$$\frac{|MFeatureValue(f, P_1) - MFeatureValue(f, P_2)|}{|Domain(f)|}$$

The qualitative version of $FDist(f,P_1,P_2)$ is defined as follows, where $V_i = MFeatureInsts(f,P_i)$.

$$\left| \frac{1}{|V_1|} \times \sum_{v \in V_1} \theta_f^{-1}(v) - \frac{1}{|V_2|} \times \sum_{v \in V_2} \theta_f^{-1}(v) \right|$$

Role Plan Distance

Role distance for a pair of plans is defined in terms of a measure of distance between the sets of fill values for the roles that the two plans share. The distance between sets of role fill values is defined to be the ratio of values that appear in one but not both sets to the total number of unique fill values (another normalized Hamming distance). We note that when possible, it may be appropriate to employ more specialized definitions that take into account the semantics of the underlying values. Such a definition could, for instance, reflect the fact that two airplanes of the same type are 'closer' than an airplane and a helicopter.

Definition 6 [Role Plan Distance] The *role distance* between plans P_1 and P_2, denoted by $RoleDist(P_1, P_2)$, is defined as follows, where $R^{\cap}$=*Roles*(P_1) ∩ *Roles*(P_2).

$$RoleDist(P_1, P_2) = \frac{1}{|R^{\cap}|} \times \sum_{r \in R^{\cap}} HammingDist(RoleFills(r, P_1), RoleFills(r, P_2))$$

Metatheoretic Plan Distance

Using the above definitions, we define the *metatheoretic distance* between two plans.

Definition 7 [Metatheoretic Plan Distance] The *metatheoretic distance* between plans P_1 and P_2 is defined as follows, where $w_t + w_m + w_r = 1$:

$$\begin{aligned} PlanDistance(P_1, P_2) = & \; w_t \times TFeatureDist(P_1, P_2) \\ & + w_m \times MFeatureDist(P_1, P_2) \\ & + w_r \times RoleDist(P_1, P_2) \end{aligned}$$

Definition 7 assumes a set of weights $\{w_t, w_m, w_r\}$ that can be set to adjust the relative importance of task features, method features, and roles in the distance calculation. Similarly, the definitions for method feature, task feature, and role distance can easily be extended to incorporate weights for individual features and roles. These weights could be set for an entire domain, or customized by an individual user on a situation-by-situation basis.

Plan Distances for the Test Suite

The motivation for defining plan distance was to support the identification of semantically disparate plans. The results in Figure 10 show that our definition is effective for

Plan Distance		Task Feature Distance	Method Feature Distance	Role Distance	Plan1	Plan2
w_t=1/3 w_m=1/3 w_r=1/3	w_t=0 w_m=.5 w_r=.5					
.07	.06	.11	.03	.08	*large-a*	*large-b*
.09	.14	0	0	.28	*large-a*	*large-c*
.12	.18	0	0	.35	*tiny-a*	*tiny-b*
.15	.17	.11	.03	.3	*large-b*	*large-c*
.15	.23	0	.15	.31	*small-a*	*small-b*
.17	.19	.14	.12	.26	*small-a*	*medium-a*
.21	.25	.14	0	.49	*small-a*	*medium-b*
.26	.32	.14	.27	.37	*small-b*	*medium-a*
.28	.36	.14	.15	.56	*small-b*	*medium-b*
.31	.34	.25	.14	.53	*medium-a*	*medium-b*
.35	.40	.25	.12	.67	*small-a*	*large-a*
.35	.41	.22	.07	.75	*medium-b*	*large-b*
.35	.20	.67	.21	.18	*tiny-b*	*small-b*
.36	.42	.25	.12	.72	*small-a*	*large-c*
.37	.39	.33	.08	.7	*small-a*	*large-b*
.40	.47	.25	.27	.67	*small-b*	*large-c*
.40	.44	.33	.18	.69	*medium-a*	*large-a*
.40	.48	.25	.27	.68	*small-b*	*large-a*
.41	.45	.33	.18	.71	*medium-a*	*large-c*
.41	.42	.4	.14	.69	*medium-a*	*large-b*
.41	.45	.33	.11	.79	*medium-b*	*large-a*
.42	.46	.33	.23	.69	*small-b*	*large-b*
.42	.46	.33	.11	.81	*medium-b*	*large-c*
.44	.33	.67	.21	.45	*tiny-a*	*small-b*
.47	.38	.67	.29	.46	*tiny-a*	*small-a*
.47	.38	.67	.29	.46	*tiny-b*	*small-a*
.50	.40	.71	.29	.51	*tiny-b*	*medium-b*
.51	.41	.71	.29	.53	*tiny-a*	*medium-b*
.52	.42	.71	.42	.42	*tiny-a*	*medium-a*
.52	.42	.71	.42	.42	*tiny-b*	*medium-a*
.55	.46	.75	.33	.58	*tiny-a*	*large-c*
.56	.47	.75	.33	.6	*tiny-a*	*large-a*
.56	.47	.75	.33	.6	*tiny-b*	*large-a*
.57	.48	.75	.33	.62	*tiny-b*	*large-c*
.58	.49	.78	.37	.6	*tiny-a*	*large-b*
.58	.49	.78	.37	.6	*tiny-b*	*large-b*

Figure 10. Distances for the SOF Test Suite

the SOF domain. The figure displays plan distances in our test suite for two cases: (a) equal weighting of task feature, method feature, and role distances (i.e., w_t=w_m=w_r=1/3), and (b) method feature and role distances only, each with weight .5 (i.e., w_t=0, w_m= w_r=.5).

For the equal weighting case, the figure shows that the 'closest' plans correspond to core plans and their variants. In particular, the shortest plan distances are between the two tiny plans (.12), between the various large plans (.07, .09, .15), and between the two small plans (.15). The distance between the two medium plans is appreciably higher (.31); as noted in Figure 1 and made apparent in Figure 8, these two plans are not simple variants of each other but rather contain key strategic differences. In addition, the plans that are farthest apart (the tiny vs large plans) are indeed the plans with the most significant variations. These results thus provide preliminary validation that the metatheoretic methods capture meaningful similarities and differences among plans.

Although the large and tiny plans are the most distant, it should be noted that the method feature and role distance measures explicitly factor out differences in plan size through their focus on method features and roles that are common to both plans. In contrast, task feature distance can be sensitive to differing plan sizes, as it measures the proportion of features that occur in only one of the two plans. The plan distance values for the case w_t=0, w_m= w_r=.5, which ignores task distance, differ somewhat from the equal weighting case. As expected, plans of similar size tend to have slightly larger distances than in the equal weighting case, while plans of quite different sizes tend to have slightly smaller distances. The overall distribution though, is broadly similar.

Discussion

To date, research on general-purpose plan summarization and comparison methods has focused on approaches that analyze plan structures and planning processes directly. For example, [Mellish & Evans 1989] generate a textual description of a plan that references every plan element, without regard to their relative importance, thus making it difficult to understand the essence of large plans. [Young 1999] improves on that work by rating the importance of an action in a plan by counting the number of its incoming causal links; only actions with certain numbers of links are included in the plan summary.

Such syntactic approaches do not necessarily shed light on the semantic content of a plan. In particular, it is possible to have plans with significant variations in syntactic structure that are semantically similar; as well, plans with similar syntactic structure may have semantic differences that are extremely significant.

One key benefit of our metatheoretic approach to plan summarization and comparison is its emphasis on semantic rather than syntactic characteristics of plans. Thus, our comparison of metatheoretic properties grounds the results in concepts that are significant from a semantic perspective, rather than an automated plan synthesis perspective.

A second benefit of our approach is the ease with which it can be customized to different users and contexts. This customization can be achieved by selecting the features and roles to drive the summarization and comparison, and by adjusting weights in the distance calculations.

Our plan summarization and comparison methods avoid domain-specific algorithms or bodies of knowledge that would limit their applicability. One problem with such general-purpose methods is that their generality often comes at the cost of depth. This tradeoff applies to our approach, in that more precise quantitative analysis tools could be developed for an individual domain that provide deeper summarization and comparison capabilities.

Our methods for plan comparison and summarization are not intended to eliminate the need for more discriminating tools. Rather, we envision the metatheoretic approach being valuable in the early stages of planning, both in terms of enabling a user to quickly grasp the main features of a plan, and to perform an inexpensive analysis of what differentiates alternative candidate plans. After developing some preliminary understanding of the plan space, a user may then wish to perform more expensive and time-consuming quantitative analyses to assess plans in detail.

A well-designed domain metatheory is critical to our plan summarization and comparison methods. We believe that the roles, task features, and method features of our metatheory are straightforward to model and should be a by-product of a principled approach to domain formulation. Although presented here as a stand-alone body of knowledge, ideally the metatheory would draw on predefined ontologies of roles and features.

The concept of roles has been considered extensively in the data modeling, knowledge representation and linguistics communities [Steimann 2000]. Our metatheory employs a basic model of role as an explicitly named relationship between an event and an entity. Our use of task features to model activity types is analogous to classes in a task taxonomy [Gil 2005]. The *background theories* of [Alterman 1988], which were developed to support plan adaptation, include concepts analogous to our notions of task feature and role. Method features have no analog to other concepts of which we are aware.

The utility of our methods when applied to larger plans could be improved by introducing a capability to localize summarization and comparison to meaningful *contexts* within a plan. The hierarchical structure of HTN plans provides one natural basis for defining contexts, namely subplans in the plan hierarchy. Identifying the conditions under which a given subplan constitutes a useful context for plan summarization and comparison presents an interesting challenge for future work.

Indexing schemes for retrieval in case-based planning, like our work, seek to provide an abstract characterization of key plan elements [Spalzzi 2001]. Case-based indexing techniques focus on similarity measures for the goal and initial state of plans in a case library; in contrast, our techniques focus on summarizing plan content. For many planning problems, the goal and plans are sufficiently disparate that very different techniques apply. *Variant process planning* presents an interesting exception in which goals consist of complex artifacts to be manufactured and retrieval involves finding process plans for similar designs [Zhang and Alting 1994]. *Group technology* (GT) is a common technique for indexing process plans that involves identifying key design and manufacturing attributes for the artifact, such as component type or material. GT can be viewed as a summarization methodology for physical artifacts that is analogous to our metatheoretic approach to plan summarization.

Conclusions

We defined an approach to plan summarization and comparison that builds on the notion of a domain metatheory. Our approach has the benefit of framing summaries and comparisons in terms of high-level semantic concepts, rather than low-level syntactic details of plan structures and derivation processes. We defined a set of techniques that instantiate this approach and evaluated them within a rich special operations planning domain. The evaluation showed that the techniques are effective in helping a user understand subtle aspects of individual plans, importance differences among plans, and the structure of the overall solution space.

Although defined for HTN planning, our metatheoretic techniques for plan summarization and comparison could be adapted readily to operator-based planning. We will be exploring that direction in future work, as well as considering how this style of technique could be used to summarize and compare temporal properties of a plan.

Acknowledgments. This work was supported by NASA grants NCC 2-1267 and NNA04CL08A. The author thanks Mabry Tyson for his assistance in modifying PASSAT to support the summarization and comparison capabilities, Peter Jarvis for developing the suite of SOF test plans used in the evaluation, and Michael Wolverton for discussions on explanation technology.

References

Alterman, R. 1988. Adaptive Planning. *Cognitive Science*, 12(3).

Bresina, J., Jonsson, A., Morris, P. and Rajan, K. 2005. Activity Planning for the Mars Exploration Rovers. In Proc. of ICAPS-05.

Erol, K., Hendler, J., and Nau, D. 1994. Semantics for Hierarchical Task-Network Planning. Tech. Report CS-TR-3239, Computer Science Dept., University of Maryland.

Gil, Y. 2005. Description Logics and Planning. *AI Magazine*, 26(2).

Mellish, C. and Evans, R. 1989. Natural Language Generation from Plans. *Computational Linguistics*, 15(4).

Myers, K.L. 1996. Strategic Advice for Hierarchical Planners. In Proc. of the 5th Intl. Conf. on Principles of Knowledge Representation and Reasoning.

Myers, K.L. and Lee, T.J. 1999. Generating Qualitatively Different Plans through Metatheoretic Biases. In Proc. of AAAI-99, AAAI Press, Menlo Park, CA.

Myers, K.L., Tyson, W.M., Wolverton, M.J., Jarvis, P.A., Lee, T.J., and desJardins, M. 2002. PASSAT: A User-centric Planning Framework. In Proc. of the 3rd Intl. NASA Workshop on Planning and Scheduling for Space, Houston, TX.

Spalzzi, L. 2001. A Survey on Case-Based Planning. *Artificial Intelligence Review*, 16(1).

Steimann, F. 2000. On the Representation of Roles in Object-oriented and Conceptual Modelling. *Data and Knowledge Engineering*, 35(1).

Tate, A., Dalton, J., and Levine, J. 1998. Generation of Multiple Qualitatively Different Plan Options. In Proc. of Artificial Intelligence Planning Systems.

Wick, M. and Thompson, W. 1992. Expert System Explanation, *Artificial Intelligence*, 54(1-2).

Young, R. M. 1999. Using Grice's Maxim of Quantity to Select the Content of Plan Descriptions. *Artificial Intelligence*, 115(2).

Zhang, H.-C. and Alting, L. 1994. *Computerized Manufacturing Process Planning Systems*. Springer.