META-PLANNING: REPRESENTING AND USING KNOWLEDGE ABOUT PLANNING IN PROBLEM SOLVING AND NATURAL LANGUAGE UNDERSTANDING

by

Robert Wilensky

Memorandum No. UCB/ERL M80/33

5 August 1980

META-PLANNING: REPRESENTING AND USING KNOWLEDGE ABOUT PLANNING IN PROBLEM SOLVING AND NATURAL LANGUAGE UNDERSTANDING

by

Robert Wilensky

Memorandum No. UCB/ERL M80/33

5 August 1980

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

META-PLANNING: REPRESENTING AND USING KNOWLEDGE ABOUT PLANNING IN PROBLEM SOLVING AND NATURAL LANGUAGE UNDERSTANDING

by

Robert Wilensky

Memorandum No. UCB/ERL M80/33

5 August 1980

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Acknowledgments

Some of the ideas in this paper arose in a graduate seminar in Artificial Intelligence and natural language processing at Berkeley in the winter of 1979. The author wishes to thank Steve Rosenberg and Joe Faletti for several valuable discussions and for their comments on various drafts of this paper.

Meta-planning: Representing and using knowledge about planning in problem solving and natural language understanding*

Robert Wilensky

Computer Science Division
Department of EECS
University of California, Berkeley
Berkeley, California 94720

Abstract

This paper is concerned with those elements of planning knowledge that are common to both understanding someone else's plan and creating a plan for one's own use. This planning knowledge can be divided into two bodies: Knowledge about the world, and knowledge about the planning process itself. Our interest here is primarily with the latter corpus. The central thesis is that much of the knowledge about the planning process itself can be formulated in terms of higher-level goals and plans called meta-goals and meta-plans. These entities can then be used by the same understanding and planning mechanisms that process ordinary goals and plans; however, the meta-planning knowledge now enables these mechanisms to handle much more complicated situations, and in a quite uniform manner.

Systems based on meta-planning would have a number of advantages over existing problem solving and understanding systems. The same knowledge could be shared by both a planner and understander, and both would be able to handle complex situations elegantly. In addition, in planning, the use of meta-planning has several advantages over more traditional methods involving constraints or critics. Meta-planning allows the full power of a problem solver to be applied to situations that are generally amenable only to special purpose processing. In addition, meta-planning facilitates the representation of some situations that are difficult to express otherwise. We have begun to introduce meta-planning knowledge into two systems: PAM, a story understanding program, and PANDORA, a problem solving and planning system.

---

*Research sponsored in part by the National Science Foundation under grant MCS79-06543.

Table of Contents

1.0 INTRODUCTION

This paper is concerned with the problems of problem solving and understanding. These problems are related because both problem solvers and understanders make use of planning knowledge to perform their respective tasks: A problem solver may generate a plan as the solution to some problem, while a natural language understander must apply knowledge about people's goals and plans in order to make the inferences necessary to explain the behavior of a character in a story (Wilensky 1978). While a story understander is quite different from a planner, both must embody a theory of planning knowledge.

I have been developing such a theory in the construction of PAM (Plan Applier Mechanism), a story understanding program. This paper is concerned not with the understanding mechanism itself, but with that part of its planning knowledge which is independent of whether that knowledge is used to explain someone's behavior or to generate a plan for one's own use. We are currently attempting to use the same theory of planning knowledge upon which PAM is based to construct a problem solving system.

This planning knowledge can be divided into two bodies: Knowledge about the world, and knowledge about the planning process itself. Our interest here is primarily with the latter corpus. The central thesis is that much of the knowledge about the planning process itself can be formulated in terms of higher-level goals and plans (meta-plans). These entities can then be used by the same sort of mechanisms that are needed for processing ordinary goals and plans; however, the meta-planning knowledge now enables these mechanisms to handle much more complicated

situations involving multiplicities of goals, and in a quite uniform manner.

1.1 A Comparison Of Problem Solving With Understanding

Before we develop this idea, it is necessary to compare problem solving and understanding in some detail. Problem solving programs often have as their goal the construction of a plan whose execution will bring about a desired state. The domain over which plan construction is performed varies considerably, involving for example, robots finding their way through rooms (Fikes and Nilsson 1971, Sacerdoti 1974), "missionary and cannibals" type problems (Newell and Simon 1972), electronic circuit design (McDermott 1977) and program construction (Rich and Shrobe 1976, Barstow 1977). The domain over which problem solving can be performed may happen to involve natural language; for example, participating in a conversation and producing an utterance have been viewed as problems in plan construction, where the problem is to create an utterance that would satisfy goals involving the transmission of certain contents or intentions (Bruce 1975, Perrault, Allen, and Cohen 1978)

Thus while problem solving may involve plan construction over quite different domains, including some linguistic ones, the essential nature of the task is the same: Given a goal, create a plan to satisfy it. In contrast, in natural language understanding, quite a different application of plans is found. Here, a reader needs to follow the goals and plans of characters in a text in order to make inferences (Wilensky 1978). Rather than actually create a plan, an understander must be able

to use knowledge about plans to understand the plan under which someone else is operating.

While problem solving and understanding both heavily involve the use of plans, it is important to emphasize how different these processes are. Problem solving has often been abstractly characterized as searching through a solution space for an answer to some problem. That is, given a goal, a set of operators, and an initial state of the world, the task is to construct a sequence of operators that transforms the initial state into the goal state. Understanding generally involves just the opposite in terms of what is known and what must be computed. Stories often state that a character took some action, from which the reader must infer why he did so and what things must have been the case for the action to have been taken. That is, the reader is given the "solution", in problem solving terminology, and must reconstruct the goal and state of the world from it. Rather than searching through solution space for an answer, an understander searches through "explanation space" to find a set of circumstances that would explain a character's behavior.

This characterization of planning and of understanding is actually somewhat flawed, as we will discuss later on. However, it serves to point out that while explanation-finding is a relative of problem solving, it is not entailed by it or dependent upon it. That is, having a good story understander does not require it to be a problem solver, nor does the existence of a problem solver fulfill the requirements of an explanation mechanism. The tasks that arise in understanding impose different requirements than do any involved in problem solving; some

simply have no correlate.

For example, the understanding task may require that a goal be inferred from the occurrence of an action, and that the existence of a goal be explained. If an understander were told that John proposed to Mary, the understander would need to infer that John probably has the goal of marrying Mary, and that this goal might have come into existence because John loved her. Problem solvers do not have the need or the facility to infer goals, or explain where they come from - their purpose is simply to act on goals given them from on high. A problem solver told that John wants to marry Mary might deduce that he should try proposing to her. But it could not explain where this goal came from nor could it infer that John has the goal from some action he undertook. Furthermore, the plan that a character chose to use might be an unusual one, one that the reader of the story would or could not produce for the same situation. The reader must therefore be able to comprehend a character's behavior in terms of a plan that it would never produce itself.

1.2 Knowledge In Common Between The Planner And The Understander

Thus understanding and problem solving are quite different, possibly inverse, processes. While each requires different programs and functions, it would be a mistake to ignore the important commonalities that these processes do have. In particular, it would seem that they should have a great deal in common in terms of the knowledge about planning that they each require. One part of this planning knowledge is essentially world knowledge. This includes a classification of

intentional structures into elements like plans, goals, and themes (Schank and Abelson, 1975), a description of the structure of these elements (e.g., plans have preconditions and actions that instantiate them, plans are used to achieve goals, etc.), and an actual body of knowledge about particular elements (e.g., asking for something is a way of getting something from someone). This point is developed at some length by Rieger (1975).

As an example, consider the fact that having possession of a book is a precondition for reading it. In problem solving, this fact would be useful if the planner had already decided that it wanted to read some particular book. Checking to see if the preconditions of this plan were fulfilled, the planner would need to consult the fact above. Learning that possessing the book is a precondition for its intended action, the planner would check to see if that condition held, and if not, would attempt to make it so. That is, the planner would establish the subgoal of achieving this precondition and then try to construct a plan to fulfill it.

An understander might use the same fact in the following manner: Given that it learned that someone had the goal of possessing a book, the understander would try to explain why this person had this goal. One kind of explanation for having a goal is that the state it is directed at is a precondition for some plan for some higher-level goal. The understander would therefore check to see if it knew of any plans for which possessing a book is a precondition. Finding the fact stated above, the understander could infer that reading the book is such a plan. The understander could then hypothesize reading a book as an

explanation for wanting to possess one.

Thus both planning and understanding make use of the identical fact, although they need to index it in different ways: in planning, one needs to get from an operator to its preconditions; in understanding, one needs to get from a precondition to that for which it is a precondition. The point here is that it is a bad idea to bury away such facts within the bowels of an understander or a planner so that they each require separate copies and possibly even separate representations of the same information. Not only is this inelegant and bad economy, it misses the point that such facts are not really "understanding" knowledge or "problem solving" knowledge. They are merely facts about the world that these processes (and possibly lots of others) happen to find useful for their particular tasks.

In actuality, many problem solvers and understanders do represent such facts declaratively. This at least potentially allows these facts to be shared by another process, although the representations used may be biased one way or another to facilitate the particular task for which the knowledge base was designed (one exception to this was a previous version of my own program, PAM, which had most of its knowledge procedurally encoded. The current version encodes its knowledge declaratively in an associative knowledge base, for the reasons being developed here as well as for other knowledge engineering advantages).

However, there is a second body of knowledge related to planning. This is knowledge about the planning process itself. This body is also required by both problem solvers and understanders, and should be shared by them. However, existing systems do not encode this knowledge

explicitly, or in a declarative and sharable form. My claim is that it is just as natural and important to have an explicit and sharable representation of this knowledge as it is for world knowledge. Moreover, such a formulation of this knowledge suggests improvements in the design of both understanders and planners.

The exact nature and need for the declarative meta-knowledge I refer to becomes apparent when one considers planning and understanding in situations involving multiple goals. By multiple goals I refer to cases in which a planner or story character is simultaneously trying to satisfy a number of goals at once, or in which there are a number of planners present, each with their own plans and goals. It is the interactions between these intentional elements that cause much of complexity in both understanding and planning.

For example, consider the following stories:

(1) John was in a hurry to get to Las Vegas, but he noticed that there were a lot of cops around so he stuck to the speed limit.

(2) John was eating dinner when he noticed that a thief was trying to break in to his house. After he finished his dessert, John called the police.

In (1), a plausible plan to achieve John's goal is to speed, but John chose to abandon this goal instead. What's needed to understand this story is not just knowledge about cops and speeding tickets, but knowledge that a person might abandon one goal if it conflicts with another goal he considers to be more significant.

Likewise, (2) strikes most people as strange since John should have reacted to the intruder more strongly. The unusualness of this story is due not to knowledge about the plans and goals involved, but the apparent unproductive scheduling of these plans. A more intelligent planner would have dealt with the threat immediately, and then perhaps returned to his meal when that situation had been disposed of.

Thus to understand the behavior of a character, or to generate an intelligent plan, it is necessary to take into account the interactions between goals. The previous version of PAM handled understanding such situations through the use of special mechanisms and knowledge to detect various kinds of goal interactions. Thus PAM had a packet of knowledge whose sole function was to spot conflicts between a character's goals, a mechanism to detect adverse interactions across the goals of different characters (goal competition), etc. The possible actions a character might take in such a situation were handled similarly.

There are a number of ways in which this solution is unsatisfactory. For one, these mechanisms for goal conflict detection, etc., were pretty much unrelated to the general explanation-finding behavior of the program when it tried to generate explanations involving just a simple goal structure. That is, PAM's normal processing was driven by the need to find an explanation for an event. It would therefore hypothesize plans that might underlie an action, hypothesize goals that might have spawned a plan, etc. until it had found a possible explanation for the input. However, detection of goal conflicts, etc., did not fit into this model at all. It was simply a mechanism off to the side that performed a function PAM would later

need, but which was not at all integrated into the basic explanation function.

While this objection may appear to be primarily aesthetic, recall that the ultimate function of detecting a goal conflict in PAM is to state an explanation for a subsequent action. For example, if in a version of story (1) we learned that John had purchased a radar detection device, the proper explanation would be that John intended to speed but didn't want to get caught. This explanation is coherent only as a plan to resolve the goal conflict between getting to a place quickly and not getting a ticket. That is, a plan to get somewhere quickly is to speed, and a way to not get a ticket is to avoid speeding. Buying a radar detection device is a plan not directed at either goal per se, but at addressing the conflict between the goals. The problem here is that determining that an action is a way of resolving a goal conflict is a form of explanation; yet it has no structural or procedural similarity to simple PAM explanations.

Another problem that this formulation poses for PAM is shared by planning programs as well. Various planning programs (e.g., Sussman 1975, Sacerdoti 1977) deal with some forms of goal interactions by providing specific programmed mechanisms that are germane to particular situations (Actually, they deal almost exclusively with conflicting subgoals and constraint violations. They are not concerned with the more general cases of goal conflict. Nor do they address the problems of goal interactions in cases in which they must contend with other competitive planners. Other such limitations will be examined more fully below). For example, Sussman's HACKER has a celebrated critic

that knows about goals clobbering "brother goals", and detects this bug in plans suggested by the plan synthesizer.

The difficulty with this type of solution is that it once again buries knowledge about how to plan in a procedure. This assures that such knowledge cannot be shared by a program that wishes to use this knowledge to understand someone else's behavior in a complicated situation. For example, if a planning program where given the goal of getting somewhere fast, it might use a critic to determine that this plan violated a "don't get ticketed" constraint, and possibly even to suggest that a radar detection device be acquired. However, as this knowledge would be procedurally embedded in the critic, it could not be used by an understander to explain why someone else was behaving in accordance with this same reasoning.

In addition, there are drawbacks for this approach within the problem solving world itself. When knowledge is embedded inside a critic, it becomes difficult to reason about this knowledge in a very general way. For example, if the resources of the critic to resolve a conflict are exhausted, it may still be possible to come up with a novel solution if the goal conflict itself were stated as a formal problem and handed to the general problem solver. But building all knowledge into a critic precludes reasoning about that knowledge in a very general manner.

The knowledge involved in the previous examples is of a somewhat different character than more mundane world knowledge. For example, it involves facts like the following:

If a person has conflicting goals, then that person may try to resolve the goal conflict.

That is, a planner must know that if it has a goal conflict, then it should have the goal of resolving it; an understander must know that if it a character has a goal conflict, then that character may have the goal of resolving it. This is not so much knowledge about the world (as "state X is a precondition for action Y" would be), but knowledge about planning itself. Such knowledge is now usually procedurally encoded in critics or special conflict detection mechanisms. But to obtain the advantage of shared knowledge and general reasoning capabilities mentioned above, this knowledge must be explicitly and declaratively formulated.

Note also that if we formulate this knowledge explicitly, it is possible to do away with the aesthetic problem of PAM's separate mechanism for dealing with goal interactions. If PAM explicitly knew that people had the goal of resolving conflicts, then detecting a conflict would be part of the more general problem of determining that a character had a goal. PAM must have such a mechanism anyway, so that it can infer that someone who is hungry is likely to have the goal of satisfying that hunger, or that someone who loves someone else may have the goal of preventing them from harm. Explaining a character's attempt at resolving a goal conflict would simply be an instance of interpreting the action as the execution of a plan aimed at the goal of resolving a goal conflict. This is exactly PAM'S ordinary explanation algorithm. It now applies to the complex situation because resolving a goal conflict is now viewed as just another goal to be achieved.

In sum, we have taken the following position:

1. We have given a number of reasons that knowledge involved in planning and understanding should be declaratively represented whenever possible. In addition to all the standard knowledge engineering arguments for declarative representations, in the particular case of planning knowledge, an important advantage of declarative representations is that they allow the sharing of knowledge between a planner and an understander. This is desirable since shared knowledge avoids duplication, eliminates problems of multiple representational forms, and simplies extending both systems. Moreover, it pays homage to the fact that human problem solvers and understanders are not separate beings. A person probably does not learn most facts just for the purpose of understanding a story or solving some problem; rather, knowledge is accumulated through diverse kinds of experience and only later applied to problem solving or story understanding.

2. While knowledge about the world is kept declaratively in many systems, knowledge about the planning process itself is usually procedurally encoded.

3. However, all the reasons for expressing ordinary world knowledge declaratively also apply to planning meta-knowledge: It is used both in understanding and in problem solving, and thus it should be sharable by the two processes for all the advantages given above. Expressing this knowledge declaratively allows a system to use this knowledge in general

reasoning processes, rather than restricting its use to the particular functions embedded in a specific program or critic. In addition, when this knowledge is cast declaratively, certain uniformities emerge. For example, in understanding, detecting a goal conflict and explaining an attempt at its resolution becomes a specific instance of the general problem of determining that a character has a goal and that an action is part of a plan to achieve a goal. No special mechanisms are required, and one theory of explanation applies both to the simple and more complicated cases.

4. Meta-knowledge about the planning process takes the form of higher-level goals, particularly those expressing the goals of the planning process itself in situations involving multiple goals and plans. We will develop this point in detail below.

2.0 META-PLANNING

This body of knowledge about the planning process is called meta-planning knowledge. By this I mean that knowledge about how to plan should itself be expressed in terms of a set of goals for the planning process (called meta-goals), and a set of plans to achieve them (meta-plans). The idea is that by expressing this knowledge in terms of goals and plans, the same planning mechanism (plan understander) that is used to produce a plan (explanation) for simple situations can operate in the more complex domain of goal interactions.

For example, consider the following situation, either from the point of view of plan understanding or plan generation:

(3) John's wife called him and told him they were all out of milk. He decided to pick some up on his way home from work.

An intelligent planner could come up with John's plan, assuming it knew that it passes by a grocery store on the route home. In order to produce this plan, it is necessary to go through the following processes:

1. Realizing that the goal of getting home and getting some milk are overlapping, i.e., that they should be pursued together rather than independently.

2. Adjusting one's plans accordingly. In this case, the plan is modified so as to

    1. Produce a route that takes the planner near the grocery store.

    2. The "go home" plan is suspended at the point at which the grocery store is reached.

    3. The "get milk" plan is executed.

    4. The "go home" plan is resumed.

Two facts of the meta-knowledge variety are involved in this example. First, one needs to know that it is generally desirable to do no more than is necessary to achieve a goal. This fact is needed in realizing that the two goals in the situation should be pursued together. This fact is a piece of meta-knowledge because it is not a fact about how to achieve a particular goal. Rather, it is based on a very general fact about the goals of the planning process, namely, that executing unnecessary steps should be avoided.

The second place in which meta-knowledge is needed is in splicing together the plans for the two goals to create a single plan. The fact used here is that it is possible to make some saving if two plans involving similar actions are executed together. This is a general fact about how to manipulate plans, not a particular fact about a specific plan or goal.

In terms of meta-planning, this situation has the following structure: The fact that a planner normally tries to avoid unnecessary steps is the meta-theme "Don't Waste Resources" (A theme is defined as something that gives rise to a goal, usually in a given situation; a meta-theme is a theme that gives rise to a meta-goal. One can think of themes (and meta-themes) as "being around" all the time, while goals (and meta-goals) come into existence only as specific entities). In this case, the situation of goal overlap causes the "Don't Waste Resources" meta-theme to bring into existence the meta-goal "Combine Plans". This is a goal like any other, and the planner now proceeds to find a plan for it. A plan that is applicable here is the meta-plan "Integrate Plans", that is, merging two existing plans to take advantage

of their common subcomponents. The application of this meta-plan achieves the "Combine Plans" meta-goal.

2.1 Meta-planning, Constraints, And Critics

One advantage of the meta-planning approach is that the problem of how to deal with complex goal interactions can be stated as a problem to be solved by the same planning mechanism one applies to "ordinary" goals. For example, one may first try out a number of canned solutions, then some standard planning procedures, and if all else fails, try to construct a novel solution.

In addition, since declarative meta-goals motivate the creation of the plan above, they can also be used to explain the behavior of a character whom a reader observed functioning in the manner described. That is, a reader of story (3) could explain John's behavior as part of a plan to achieve two specific goals, and which at the same time avoided wasting any resources unnecessarily.

Note that there are a number of important differences between meta-planning and planning using constraints or critics. As was just pointed out, meta-planning involves the use of declarative knowledge that can be used for both understanding and planning, whereas knowledge in the form of a critic is usually procedural and therefore unsharable. Another difference is that constraints and plan generators are asymmetric in that constraints reject plans, but they don't themselves propose new ones. Generally, if a constraint is violated, a plan is rejected and it is left to the plan generator to propose a new plan. In

contrast, meta-goals accomplish what constraints are intended for by formulating a problem that the current plan entails. Unlike a constraint, the existence of a meta-goal causes the planner to try to solve a particular problem, rather than just return control to the plan generator.

Unlike constraints, critics can change the existing planning structures to eliminate a problem. However, such changes are limited to particular types of transformations known to the critic. The meta-planning approach separates the formulation of the problem from its detection and solution. Activating a meta-goal corresponds to detecting a violation, the meta-goal itself to the formulation of the problem, and finding an appropriate meta-plan to creating a solution. The advantage of this approach is that it does not require the critic to embody the solution; instead, constructing the solution is left up to the general resources of the planning process. Meta-planning allows all the benefits of having a general problem solver to be available to the task of resolving a constraint violation. Since the implicit assumption is that having a general problem solver is a good idea to begin with, then we are better off having a general problem solver at hand to handle constraint violations and the like than relegating this responsibility to an expert critic (Of course, this does not prevent us from having such expert knowledge as a critic possesses available. We are simply allowing this knowledge to interact with all other knowledge as it can now take part in general deductions).

In addition to increasing the chances of finding an answer, the meta-planning formulation also provides more flexibility when no answer is available. In general, when a critic finds fault in a plan, it tries to put a fix into effect right away. If it cannot apply one, the plan must be rejected. However, since a meta-goal represents the formulation of a problem, the existence of the problem may be dealt with beyond its being resolved. For example, the problem solver may simply decide to accept the flawed plan if the violation is viewed as not being too important. In fact, it may be the case that changing the existing plan to resolve the problem may result in other problems that are deemed even more serious. In this case, the best choice is to accept the original flawed plan. By separating solving the problem from its formulation, the problem may be accessed as opposed to treated, an option that critics do not usually leave open.

Meta-planning is also meant to cover a somewhat different scope than that covered by critics and constraints. To begin with, meta-planning knowledge is not always critical in nature. For example, it might be possible to suggest a plan for combining the normal plans for two goals as in the case of example (3) without first proposing a flawed planning structure. That is, the knowledge may productively affect the planning process without there necessarily being a bad plan around for a critic to react to. In general, meta-planning advises the planner about goal interactions and the like, and only some of this knowledge specifies situations to be avoided.

Furthermore, meta-goals are domain independent, encoding only knowledge about planning in general. In most planning systems, constraints and critics embody knowledge that is domain dependent as well as that which is not. For example, a specific critic may exist that encodes knowledge about the types of situations to avoid in a particular task domain. With meta-goals, task-specific constraints are enforced by specifying that some particular situation would activate some general meta-goal.

McDermott's notion of a policy, or a secondary task comes closest to the notion of meta-planning I propose here. A policy is essentially an explicitly represented constraint. Like meta-goals, policies have the advantage that they may easily enter into general deductions. The primary differences between a policy and a meta-goal are that meta-goals include goals that are not necessarily constraints per se; meta-goals refer only to facts about planning as their domain, whereas policies may include domain specific information; policies often entail the creation of pseudo-tasks, whereas meta-goals have meta-plans that deviate less from the structure of normal plans.

Hayes-Roth and Hayes-Roth (1978) uses the term meta-planning to refer to decisions about the planning process. While my use of the term is similar to theirs, they include all types of planning decisions under this name, and their meta-planning is not formulated in terms of explicit meta-goals and meta-plans. I use the term to refer only to a subset of this knowledge, and only when that knowledge is expressed in terms of explicit meta-goals and meta-plans.

2.2 Kinds Of Meta-goals

The following is a brief description of the more important meta-goals so far encountered, along with the meta-themes and situations in which they arise, and some of the meta-plans applicable to them. This list is not meant to be complete. It merely reflects the current state of our analysis.

Situations, Meta-themes, Meta-goals, and Meta-plans

1. Meta-theme - Don't Waste Resources

   Situations to Detect

   1. Goal Overlap

      Meta-goals initiated:

      1. Combine-Plans

         Associated meta-plans:

         1. Schedule Common Subgoals First
         2. Plan Integration
         3. Plan Piggybacking (find a new plan that simultaneously fulfills both goals)

   2. Goal Concord

      Meta-goals initiated:

      1. Ally-Plans

         Associated meta-plans:

         1. Divide Task
         2. Piggyback Goal (Try to capitalize on the plan of another planner to fulfill one's own goal).

3. Multiple Planning Options (more than one plan is applicable to a known goal)

   Meta-goals initiated:

   1. Choose-Least-Costly-Scenario

      Associated meta-plans:

      1. Simulate-and-Select (Determine the cost of the various options and pick the least costly one)

4. Recurring Goals (A goal that arises repeatedly)

   Meta-goals initiated:

   1. Establish-Subsumption-State (Establish a state that fulfills a precondition for a plan for the goal and which endures over a period of time (see Wilensky, 1978b))

      Associated meta-plans:

      1. Plans are a function of the subsumption state to be achieved.

2. Meta-theme - Achieve As Many Goals As Possible

   Situations to detect

   1. Goal Conflict

      Meta-goals initiated:

      1. Resolve-Goal-Conflict

         Associated meta-plans:

         1. Various "canned" plans for specific kinds of goal conflicts

         2. Obtain-More-Resources (if the conflict is based on a resource shortage)

         3. Change-Circumstances (if the conflict is based on invoking a preservation goal in the course of planning for another goal)

4. Select-New-Plan (if the conflict is cause by an adverse plan interaction, try considering another plan)

5. Reschedule-Tasks (e. g., see if the order in which steps were scheduled is causing the problem)

6. Other general forms of goal conflict resolution are dependent on the particular type of goal conflict, and are described in Wilensky (1978).

3. Meta-theme - Maximize the Value of the Goals Achieved

Situations to detect

1. Unresolvable Goal Conflict

Meta-goals initiated:

1. Choose-Most-Valuable-Scenario

Associated meta-plans:

1. Simulate-and-Select (includes sub-plan of Goal-Modification as a way of producing alternatives to consider)

4. Meta-theme - Avoid Impossible Goals

Situations to detect

1. Circular Subgoals (A subgoal generated in a plan is the same as some goal to which it is instrumental)

Meta-goals initiated:

1. Resolve-Circularity

Associated meta-plans:

1. Goal substitution

2. Plan modification

2. Too-difficult-goal

Meta-goals initiated:

1. Form-Alliance (look for an ally with concordant goals)
2. Resolve-Need

   Associated meta-plans:

   1. Goal substitution
   2. Plan modification

In addition to these meta-themes, the following theme has a meta-knowledge quality to it, although it does not necessarily produce meta-goals per se:

Theme - Don't Violate Desirable States

Situations to detect

1. Danger

   Goals initiated:

   1. Preserve-Endangered-State

      This is strictly a preservation goal. However, these often act like meta-goals when they occur due to another of the planner's goals.

      Associated meta-plans:

      1. Prevent-Endangering-Event
      2. Change-Circumstances (Modify circumstances so that plan will not have anticipated negative effect)

2. Maintenance Time

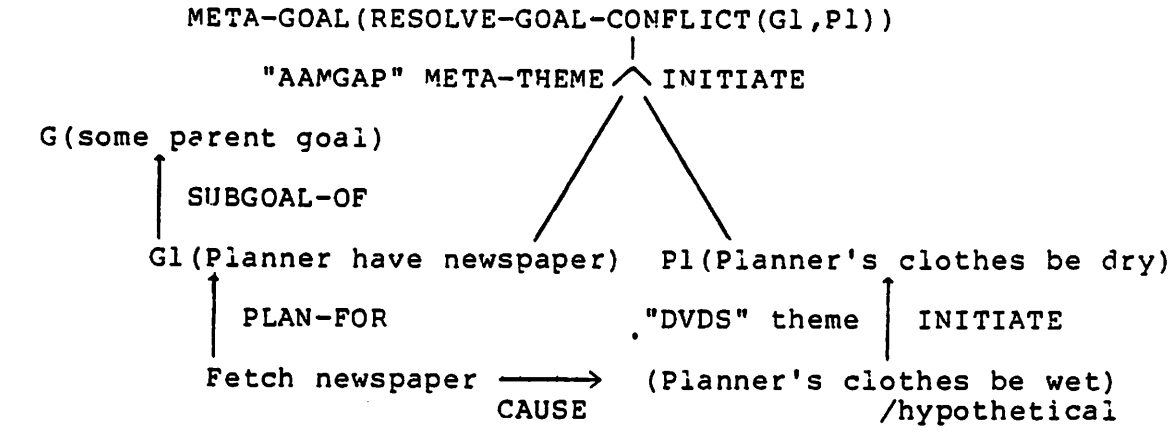
   Goals initiated:

   1. Perform-Maintenance

2.3 Example

The following example is a sketch of how meta-goals are used in the planning process. Suppose a planner were given the task of fetching a newspaper from outside. It's raining outside, however. We will assume that the planner will first consider the "normal" plan for a task if one is known. The normal plan for getting the newspaper is walking outside and carrying it back in, which in this case would cause the planner to get its clothes wet. This situation would cause the "Don't Violate Desirable States" theme to create the goal Preserve-Endangered-State(Clothes be dry) (For the time being, we will ignore the problem of just exactly how the planner finds the themes relevant to a given situation, thereby creating an appropriate goal).

Since this preservation goal came into existence as the result of some intended action by the planner itself, a goal conflict must exist between this goal and the goal from which the other action originated. Since a goal conflict threatens the fulfillment of a goal, the meta-theme "Achieve As Many Goals As Possible" causes the meta-goal Resolve-Goal-Conflict(G1,P1) to come into existence, where G1 is the goal of having the newspaper and P1 the goal of preserving the dryness of the planner's clothing. The state of the planning at this point is diagrammed in the figure below:

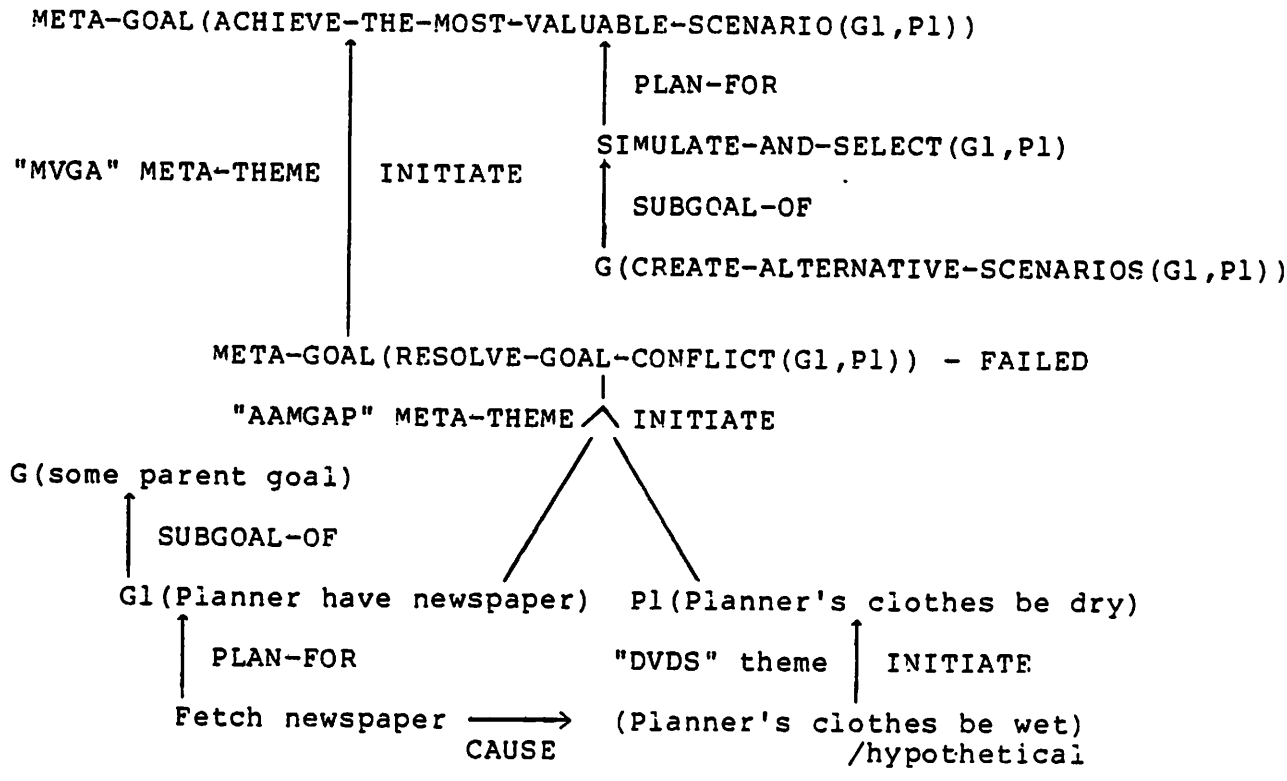


Now a plan for this meta-goal is sought. Since the meta-goal is treated just like an ordinary goal by the planner, normal plans for resolving the conflict are sought first. One specific stored plan for this situation is to wear a raincoat while performing the plan for goal G1. Suppose the planner proposed this plan, which spawns the sub-goal of acquiring a raincoat. If a raincoat were readily available, then the plan for resolving the goal conflict succeeds and the plan for G1 can be executed.

On the other hand, the subgoal of obtaining a raincoat might spawn a plan that involves going outside. As this is a circular subgoal, the "Avoid Impossible Goals" initiates a Resolve-Circular-Goals(G1,S1) meta-goal, where S1 is the subgoal just created that is identical to G1. A meta-plan applicable here is Modify-Plan, which tries to choose another plan that does not spawn subgoal S1. Suppose that this fails, and no other canned plan for achieving the Resolve-Goal-Conflict(G1,P1) can be found. Since this is treated just like any ordinary goal, the planner might try to create a more novel plan here. For example, one general strategy available for resolving goal conflicts based on

invoking a preservation goal is to alter the circumstances that enable the intended action to have its undesirable effect. For example, the planner might try interposing some object in between himself and the rain, like an old newspaper, or simply wait for the rain to stop. Both waiting and actively changing a precondition are general strategies applied to this particular situation.

Suppose that such plans are not generated or are rejected for other reasons. Then the planner has failed to fulfill its Resolve-Goal-Conflict goal. The existence of an unresolvable goal conflict indicates that some goal is about to fail, and the "Maximize the Value of the Goals Achieved" meta-theme activates the Achieve-the-Most-Valuable-Scenario(G1,P1) meta-goal. The plan for this goal is Simulate-and-Select, that is, invoke the subgoal Create-Alternative-Scenarios(G1,P1), evaluate each one, and then abandon the goals deemed least important. The situation at this point is as follows:



In this case, Create-Alternative-Scenarios has a relatively straightforward task, as there are only two scenarios: Abandoning G1, or abandoning P1 (Actually, Create-Alternative-Scenarios is also empowered with the ability to do goal modification. For example, it might suggest changing P1 to "Keep clothes as dry as possible", in which case a suitable plan might be to run out to get the paper. While the creation of these scenarios is of course more complex, we assume that they are evaluated in the manner we now describe for simpler scenarios). Now the planner has to make a judgment on the relative value of the goals involved. To make this evaluation, we assume that the planner is able to attach some sort of context independent value on each of its goals, say, for example, a value between 0 (don't care) and 10 (life or death). The value of each scenario is defined as the sum of the value

of its individual goals. If "not getting my best suit wet" is a 5 and "having the paper" a 3, the two scenarios have a relative value of 5 and 3, so P1 is selected over G1. In this case, the planner stays inside and does without his paper.

Scenarios involving goal modification are handled the same way. For example, if getting one's clothes just a trifle wet is a 2, then that scenario is valued as a 7 (since one ends up with the paper as well), and would be chosen over either of the two more polarized versions. I stress the importance of goal modification as a strategy here, as true goal abandonment seems much rarer than some form of goal modification. For example, a planner is much more likely to listen to the news on the radio, wait for the rain to stop, try to get someone else to bring in the paper, change to less valuable clothing, use something instead of a raincoat, etc., instead of abandoning his goal altogether.

In the example above, meta-planning allows the problems faced by the planner to be formulated as goals and then given to the planner to solve in a general fashion. Similarly, suppose we were trying to understand a story about someone who wanted to bring in the paper in the rain. After reading this much, a reader could infer that that person has the preservation goal of not getting his clothing wet. If the reader then learns that he decided not to go out, for example, the reader can infer the following explanation for this behavior: The person must have had a failed Resolve-Goal-Conflict between the two goals, and was now trying to fulfill the Achieve-the-Most-Valuable-Scenario meta-goal. Since the planner chose

to stay dry and abandoned getting the paper, the former must have been the goal the planner valued most. While the processing is quite different than in the planning example above, the same declarative meta-knowledge is used in both cases.

3.0 THE STRUCTURE OF A PLANNER AND AN UNDERSTANDER BASED ON META-PLANNING.

The example above illustrates the use of meta-planning in both planning and understanding. In this section, a more detailed description of my view of these processes is given.

3.1 The Understander

The design of a story understander that uses intentional knowledge has been given elsewhere (Wilensky 1978). PAM has since been completely re-implemented, primarily to give it a declarative associative data base and to handle some higher-level story structures described in Wilensky (1980). A detailed description of this implementation will appear in a forthcoming technical report. Thus only the changes implied by the meta-planning approach need to be discussed here.

Recall that one of the understander's tasks is to interpret the actions of a character in a text as part of some reasonable plan. This involves inferring that an action is part of a plan for some goal, inferring that a character has a goal, and accounting for the effects of goal interactions. These processes are driven by the need to find an explanation for a character's behavior. Thus if an understander learns that someone picked up a phone book, it might hypothesize that the

person was going to use it to find someone's phone number in order to call that person up in order to have a conversation, etc.

To make these inferences, the understander needs to know when goals arise. For example, if an understander learned that John wanted to eat, it might infer that he was hungry; if it learned he wanted to marry Mary, it might infer that he loved her or was after her money. In general, the understander must be able to detect situations that give rise to goals in order to interpret a character's behavior.

Given that the understander has the ability to hypothesize a character's goals, the introduction of meta-planning knowledge gives a rather straightforward way of dealing with more complicated goal interactions. The problem here is to interpret a character's behavior when faced with a number of interacting goals at the same time. For example, if we were told that John wanted to go out with his secretary but was afraid his wife would find out, we could explain his calling her to tell her he had to work late that night as a way of getting around this conflict. As was mentioned above, this previously required a separate process to monitor goal interactions and attempts at dealing with there effects.

Using meta-planning, however, we simply supply the understander with the description of the situations in which meta-themes cause meta-goals to arise, and with knowledge about the meta-plans associated with each meta-goal. If a goal conflict arises in a story, the understander will spot it and infer that the character has the meta-goal of resolving it just as it will infer that a character has the goal of eating if it learns that that character is hungry. For example, going

out with one's secretary gives rise to the preservation goal of preventing one's spouse from knowing. The meta-theme "Achieve As Many Goals As Possible" recognizes this situation as one leading to the Resolve-Goal-Conflict meta-goal. Thus this goal can be inferred, and subsequent actions interpreted as plans to fulfill it. In this case, telling one's wife that one has to work late might be interpreted as a normal plan for this goal.

Again, the advantage of the meta-planning approach is that nothing special is done for these more complicated situations other than to supply the understander with meta-knowledge in the form of themes, goals and plans. Recognizing a complicated situation and inferring what a character might do is reduced to the problem of knowing that a theme is active and what goals it may give rise to. Since a mechanism that does this is needed to find ordinary explanations, no additional mechanisms are warranted here.

Note that this is true even for some basic control-related issues. For example, one problem that often arises in story understanding is whether to make a "backwards" or a "forwards" inference. For example, an understander often needs to infer that John loves Mary upon learning that he wants to marry her; sometimes it needs to hypothesize that he wants to marry her upon learning that he loves her. In one case, a goal is hypothesized from a theme; in the other, a theme is inferred from the presence of a goal. In general, it is difficult to know which strategy to employ for any given situation; often, exactly what will happen is a function of the number of different things that can be predicted from a known fact, the specificity of the knowledge available

to the system, etc.

Exactly the same situation arises in the use of meta-planning for the detection of goal conflicts and other goal interactions: Sometimes the presence of a goal conflict might only be inferable from the action a character takes to resolve it; sometimes it might be predictable from the presence of two goals. The point here is that we need not commit ourselves to a particular control strategy for these cases, any more than we did for making simpler inferences. Since detecting goal interactions is now done in terms of higher-level meta-goals, the same criteria that influence the inference process for simpler cases is applicable here.

3.2 The Planner

The design of a planner based on meta-planning is now given. This planner is composed of the following major components:

1. The Goal Detector

   This mechanism is responsible for determining that the planner has a goal. The goal detector has access to the planner's likes and dislikes, to the state of the world and any changes that may befall it, and to the planner's own internal planning structures and other internal states. The goal detector may therefore establish a new goal because of some change in the environment, because such a goal is instrumental to another goal, or in order to resolve a problem in a planning structure it is creating.

2. The Plan Generator

The plan generator proposes plans for the goals already detected. It may dredge up stereotyped solutions, it may edit previously known plans to fit the current situation, or it may create fairly novel solutions. The plans may be applicable to a single goal, or to several of the goals present at once. The plan generator is also responsible for expanding high-level plans into their primitive components to allow execution.

3. The Executor

The executor simply carries out the plan steps as proposed by the plan generator. It is responsible for the detection of errors, although not with their correction. In general, this might entail establishing a new goal and creating a plan to set things right.

The only part that is really new here is the goal detector. As we mentioned above, most planners do not worry about where their goals come from; high-level goals are generally handed to the planner in the form of a problem to be solved. However, it is desirable for a robot problem solver to worry about goal detection for a number of reasons:

1. For the problem solver to be autonomous, it will need to know when it should go into action. E. g., before it solves the monkeys and bananas problem, the robot would need to recognize that it was hungry and that it should try to feed itself. If it were given a set of tasks that included "keep the nuclear

reactor from melting down", it would need to know that it should have the goal of shutting the reactor down when the dial on the control panel indicates a cooling failure.

2. An autonomous planner would also need to deal with its own preservation. It would therefore need to react to changes in its environment; for example it would need to know it should have the goal of getting out of the way of an avalanche. It should also be capable of knowing when it needs maintenance or when it should replenish its power supply. Moreover, a real planner may have to deal with adversity. For example, a robot operating a nuclear power plant might need to fend off a terrorist attempting to sabotage its activities. Other planners may interfere with its goals in a number of ways. The point is that the the robot planner trying to solve the monkeys and bananas problem may have to contend with another robot also trying to get the bananas for itself.

3. Even a non-autonomous planner would need to know when it should help. For example, a system designed for man-machine interaction would need to know when it should take over control from its human operator and when it should relinquish it. It would need to know which aspects of the problem being worked on were its responsibility so that it could assist the human in the appropriate situations without the human explicitly engaging it.

4. The planner needs to know about internally generated goals. For example, virtually all problem solvers detect that they have a subgoal. In addition, they must also know when they should be trying to resolve a goal conflict, when they should try to combine their own plans to produce a more efficient one, and when they should evaluate the relative importance of their goals. For example, the robot in charge of the nuclear reactor might have to realize that it had an unresolvable goal conflict between "output as much electricity as possible", "keep the floors clean" and "prevent a meltdown", and that it therefore needs to determine which of these is the most important to achieve.

The goal detector operates through the use of a mechanism called the Noticer. The Noticer is a general facility in charge of recognizing that something has occurred that is of interest to some part of the system. The Noticer monitors changes in the external environment and in the internal states of the system. When it detects the presence of something that it was previously instructed to monitor, it reports this occurrence to the source that originally told it to look for that thing.

The Noticer can be thought of as a collection of IF-ADDED demons whose only action is to report some occurrence to some other mechanism. In addition, demons usually monitor only insertions into the data base of a system, while the Noticer may have to monitor external changes, or the state of a systems process as well.

Goals are detected by having themes and meta-themes asserted into the Noticer with orders to report to the goal detector. When an event matching the desired specifications occurs, the goal detector can then assert the existence of some particular goal. For example, suppose the planner were in possession of the knowledge that a person who is aware that some undesirable circumstance may come about will have the goal of preventing that circumstance from happening. This knowledge could be represented as follows:

```
(INITIATE
 *AND
  *AWARE ?ACTOR ?X/HYPOTHETICAL)
  *UNDESIRABLE ?X ?ACTOR))
 *GOAL ?ACTOR *PREVENT ?X)))
```

That is, the existence of conditions described in the first clause causes the condition described in the second to come about. As usual, question marks indicate variables. The slash notation is used to express a qualification; it may be thought of as a shorthand for an additional ISA conjunct. Also, as has been pointed out by McDermott (1978) and others, the semantics of terms like "hypothetical" is highly problematic. Here I use it simply to mean that the planner has reasoned that some circumstance will come into existence at some point in the future. Similarly, the predicate "undesirable" should simply be read as "positive", i. e., this is a primitive evaluative judgment.

The first argument in this predication would be handed to the Noticer. If it were the case that "(UNDESIRABLE (DEAD ?ACTOR) ?ACTOR)" were in the planner's data base (i. e., that the planner knew that being dead was considered undesirable for oneself), and the planner later learned that (DEAD *EGO*) were imminent, the Noticer would notice that a condition it was told to watch for had been matched, and would report this occurrence back to the goal detector. The goal detector then uses the formula this condition originated from to infer the goal "(PREVENT (DEAD *EGO*))", i. e., prevent itself from dying.

Note that this formulation would cause the Noticer and goal detector to infer that someone else would have the goal of preventing their own death if that person became aware of some danger. This is a valid inference, but of course, not a goal of the planner making it. There are several ways of handling this. One is to restrict the predications by replacing the appropriate references to ?ACTOR with *EGO* so that the Noticer doesn't "false alarm" to other people's goals. Another solution is to let the system make these inferences, and then have the goal detector examine them to see if it has deduced one of its own goals. The second solution has the advantage that inferring the goals of other planners is something that usually needs to be done anyway for other purposes, and allows the same piece of knowledge to be used for both cases. Probably a mixture of both strategies would be valid, as it is plausible that people possess some knowledge that is used to infer one's own goals but not the goals of others.

When a new goal is detected, it is moved to the front of the "goal queue." This is the storage structure for currently active goals. If it was inappropriate to put this goal here, say, because some other goal is clearly more important or urgent, this will cause a meta-theme to create a re-scheduling meta-goal. This will get put on the front of the queue; we can guarantee that the urgency of such a goal is computed to be at least the urgency of most urgent goal it is re-scheduling. It will therefore get acted upon immediately and cause the unimportant goal to get demoted.

As was just indicated, the goal at the front of the queue is worked on first. That is, it is given to the planner, which tries to reduce that goal to a "task network" of plans and subgoals that eventually bottoms out in a set of primitively executable actions. Our planner consists of three components:

1. Proposer, that suggests plausible plans to try
2. Simulator, which tests plans by trying to compute what it would be like to execute them
3. Revisor, which can edit and remove certain parts of a task network upon request

Proposer begins by suggesting the most specific plan it knows of that is applicable to the goal. If this plan is rejected or fails, Proposer will propose successively more general and "creative" solutions. We will have little to say about this process here, as we place most of our emphasis on dealing with the interactions between

fairly standard plans rather than upon the creation of strikingly original ones.

Once Proposer has suggested a plan, Simulator starts computing what will happen to the world as the plan is executed. The difficult problems in conducting a simulation involve reasoning about "possible world" type situations which are not amenable to standard temporal logic (McCarthy and Hayes, 1969). However, we previously finessed this issue by defining hypothetical states in terms of what the planner thinks of in the course of plan construction. In other words, our solution is to let the system assert the changes that would be made into a hypothetical data base, in the meantime letting the goal detecter have access to these states. Thus if the plan being simulated would result in the planner dying, say, this would constitute a hypothetical undesirable state, which might trigger further goals, etc.

As the Simulator hypothetically carries out the plan, and other goals and meta-goals are detected by the goal detector, the original plan may have to be modified. This is done by explicit calls to the Revisor, which knows the plan structure and can make edits or deletions upon request. The modified plan structure is simulated again until it is either found satisfactory or the entire plan is given up and a new one suggested by Proposer.

4.0 SOME DETAILS OF META-GOALS AND META-PLANS

In this section some of the more important meta-themes, goals and plans are described in somewhat more detail. Many of these items deal with interactions between goals. In particular, the following goal relationships seem to play a significant role:

1. Goal Conflict - An adverse interaction between the goals of a planner.

2. Goal Competition - An adverse interaction between the goals of different planners.

3. Goal Overlap - A positive interaction between the goals of a planner.

4. Goal Concord - A positive interaction between the goals of different planners.

5. Goal Subsumption - Establishing a state that makes it easier to fulfill a recurring goal.

Most of these relationships have been described in Wilensky (1978). The only new one is goal overlap. This relationship is discussed in detail in the following section.

4.1 The Value And Cost Of Goals And Meta-goals

In the discussion of meta-goals that follows, it is necessary to refer to the cost of achieving a goal and the value of that goal to the planner. These issues arise, for example, in trying to decide which of two conflicting goals should be pursued, or whether a plan for a goal is worth the resources it consumes. The theory of planning presented here does not specify values for particular goals, or a particular "goal calculus" of goal value manipulation. However, it does require that the value of a goal and the cost of a plan have the following properties:

1. All goals and costs are in principle comparable, although the result of the comparison need not be definitive. That is, the planner may have to judge whether slighting a friend is worth a certain amount of money, or whether two apples is worth three oranges. One way to make such comparisons is to assign to every goal a point or a range on a numerical scale. Goals with overlapping ranges would constitute difficult decisions, whereas goals with disjoint ranges should constitute clear preferences.

2. The cost of using a resource is equivalent to the value of the preservation goal of keeping that resource. I. e., costs and values are really the same sorts of objects. Thus what we say about evaluating costs below will generally apply to evaluating goals, and visa versa, as a cost is just a negative goal value.

3. The value of a set of goals is computable from the values of its members. For example, linear summation seems to work for independent goals. That is, the value of two goals is the sum of their values; the value of a goal given a plan with a certain cost is that value minus the cost. By independent, I mean that the goals cannot be merged into a single bigger goal, although they may interact with each other. For example, the values of having a car and eating an ice cream cone is the sum of the value of each goal. But the value of eating five ice cream cones may be somewhat less than five times the value of eating one.

   Resulting values can be compared as well. That is, the degree to which one set of goals is better than another is the difference between the sums of the values of the two sets. Of course, we do not need to commit ourselves to a particular theory of how to combine the values of goals here. We need only assume that it is possible to do so.

4. The value of high-level goals can be evaluated separately and independent of context. We must assume that we can assign a value to a goal (or more precisely, to the state the goal is aimed at bringing about) just by considering that goal, and without regard to any costs incurred by a plan for the goal or to any added benefits one gets from executing such a plan. We can then assess the net value of a task network by adding together all the benefits it brings about and subtracting all its costs.

Of course, sometimes assigning a value to a goal will be difficult; sometimes only a vague indication of value is possible. For example, the value people usually attach to human life is "very high"; however, this assessment is not much help in comparing the value with other goals in the high value range. Difficulties of assigning values is one of the factors that creates difficulty for goal comparison.

5. On the other hand, the value of a sub-goal is computable from the value of the goals to which it is instrumental. In general, the value is equivalent to the sum of the values of the dependent goals. These values are not inherent, of course, as the value of a subgoal may go change if a plan involving that goal is altered.

6. The value of a meta-goal is strictly inherited from the values of the goals it refers to. For example, the meta-goal Resolve-Goal-Conflict is worth the value of the goals that would be abandoned if the conflict were not resolved.

One consequence of these assumptions is that meta-goals can be compared with ordinary goals for planning purposes. If the meta-goal Resolve-Goal-Conflict ends up conflicting with some other goal, then another Resolve-Goal-Conflict meta-goal may be created whose value is the minimum of the value of the first Resolve-Goal-Conflict meta-goal and the other goal.

This situation is actually fairly common and works out rather nicely. For example, time-based goal conflicts can often be resolved by giving up a "background goal", like eating or sleeping. However, this leads to a second conflict between resolving the first conflict and satisfying hunger or tiredness. If this conflict cannot be resolved, the planner may have to either abandon resolving its first goal conflict, or abandon a background goal. He makes this decision simply by comparing the value of the first Resolve-Goal-Conflict goal, which is the minimum of the value of its two goals, with the value of the background goal.

It might at first seem like a problem would arise here with scheduling, as by definition, the value of a Resolve-Goal-Conflict goal will usually be less than the value of the most valuable of the conflicting goals. All other things being equal, Resolve-Goal-Conflict goals would receive less attention than the more important goal in the conflict, which would subsequently be attempted before the conflict were resolved. However, a number of things prevent this from happening. In general, "Achieve as many goals as possible" motivates the scheduling of goals, so conflict resolution goals will initially get precedence over those goals whose conflict they are trying to resolve. Only when it appears that some goal is failing will "Maximize the value of goals achieved" be useful. In other words, Resolve-Conflict goals are treated like subgoals, in that they are worked on before their parent goals are looked at again.

4.2 Meta-goal Details

4.2.1 Combine-Plans

Most of the meta-knowledge concerning goal overlap falls under the "Don't Waste Resources" meta-theme. The meta-goal given rise to here is Combine-Plans, which is satisfied when part of the task networks for the overlapping goals come to be joined together. Several plans for doing this were referred to in the preceding section, including Schedule Common Subgoals First, Plan Integration, and Plan Piggybacking. The details of goal overlap are discussed in the next section.

4.2.2 Choose-Least-Costly-Scenario

This meta-goal arises in situations involving multiple planning options under the meta-theme of "Don't Waste Resources". Given a set of plans for a goal, this meta-goal is achieved when the scenario with the cheapest cost is computed. The Simulate-and-Select meta-plan is useful here. This works by first computing a number of plausible scenarios, and then calling the simulator to determine the state of the world that would exist for each one. The cost of each scenario can then be measured from the values of the resources consumed and the states that exist before and after the plan is executed. The scenario with minimum cost can then be selected. In the case of Choose-Least-Costly-Scenario, the scenarios are generated simply by considering alternative plans.

4.2.3 Choose-Most-Valuable-Scenario

Choose-Most-Valuable-Scenario is a similar meta-goal that arises when the "Maximize the Value of the Goals Achieved" meta-theme is activated by the existence of an unresolvable goal conflict. Given a set of goals in unresolvable conflict, the meta-goal is to choose the subset of them to work on that maximizes the gain to the planner. The Simulate-and-Select meta-plan is applicable here as well. In this case, the plan can suggest various goals to abandon and measures the value of the remaining goals.

In addition, Simulate-and-Select can attempt goal modification as a way of generating plausible scenarios. To do this, the plan suggests various modifications to the some goals that would allow some additional goals to be achieved. The value of these modified goal scenarios are compared along with the ordinary abandonment scenarios to obtain a reasonable choice of action.

As was noted above, goal modification seems to be the case rather than the exception. Giving up a goal altogether is unusual. For example, if there is a conflict between one's career and one's personal life, the general solution is to sacrifice the degree to which one is achieved in order to partially fulfill the other, rather than to give one up entirely.

An alternative way of dealing with goal modification is to say that the semantics of achieving a goal is really to achieve it to a degree. The difference between this formulation and the one presented so far is that it suggests that plans that partially fulfill a goal would be

proposed right along with plans for complete fulfillment; in the current formulation, these are only suggested by the planner after other plans fail. For example, in this formulation, the goal of not getting wet would be interpreted as getting as little wet as possible, whereas in the formulation given above, a modified goal has to be created to allow this possibility. In addition, this alternative formulation would suggest that goals are generally only partially rather that totally fulfilled, which is probably the case.

The problem with this formulation is that it makes it difficult to specify goals that must be achieved exactly. It also is not amenable to substitution modification rather than degree modification. For example, it might be necessary to substitute "have stool" with "have tall box" in some plan. This seems to be more easily accommodated by doing an explicit substitution than by fudging the semantics of goal fulfillment.

The best way of handling this is by forcing partial fulfillment to be explicitly expressed in a goal specification. For example, rather than specifying the goal as "not be wet", we can express it as "minimize the degree of wetness". This makes it possible to have goals that are variably satisfied, and to suggest partially fulfilling plans together with totally fulfilling ones. At the same, we can still have "discrete" goals simply by specifying them; some of these would still be amenable to goal modification.

4.2.4 Establish-Subsumption-State

This goal arises when the planner has a recurring goal that is either not subsumed, or whose current subsumption state is inadequate for some reason. For example, if John finds himself needing some tool repeatedly he might decided to buy one so that achieving this goal will be easier next time the need arises. Establishing a subsumption state first requires determining what a subsumption state is for the recurring goal involved, and then embarking on a plan to achieve this state. This latter plan is of course totally dependent on the nature of the subsumption state one must achieve.

4.2.5 Resolve-Goal-Conflict

Resolve-Goal-Conflict is invoked by the "Achieve As Many Goals As Possible" meta-theme, and occurs whenever a goal conflict is present. Note that by goal conflict, we mean to include situations in which goals are in conflict through the plans chosen for them, as well as those cases in which the goals themselves are inimicable. I distinguish between three cases of goal conflict: Those based on a shortage of resources, those bases on mutually exclusive states, and those based on invoking a preservation goal in the course of planning for another goal.

Since there are different causes of goal conflict, ranging from bad scheduling to inherently exclusive goal states, different plans are applicable in different situations. In fact, many of the plans applicable to Resolve-Goal-Conflict are "canned" plans that are specific to conflicts between particular pairs of goals. For example, as was mentioned above, a canned plan for resolving the conflict between going outside in the rain and getting one's clothes wet is to wear a raincoat.

A standard plan for resolving conflicts based on a shortage of time resources is to abandon a background goal such as eating or sleeping.

On a more general level, there are plans applicable to particular classes of conflict.

1. If the conflict is based on a scarcity of some resource, then Obtain-More-Resources is suitable. This plan will have to be further specified depending on the type of resource causing the shortage. For example, if the short resource is time, the plans for the conflicting goals can often be integrated together, thus resolving the time shortage.

2. If the conflict is based on invoking a preservation goal, then the Change-Circumstance plan is applicable. This is a general strategy for dealing with preservation goals in which the conditions surrounding the action invoking the goal are changed so that the action no longer causes an undesirable effect. For example, in the case of going out into the rain, interposing an object between the rain and oneself will change the circumstances so being in the rain no longer causes one's clothing to get wet. One particular variant here is Waiting, in which case the planner does no action in hopes that the circumstances will change themselves.

   The other general strategy for dealing with preservation goals is to prevent the event that causes the goal to arise in the first place. This is not applicable here, however, as the causing event was presumed to be instigated by the planner and

would therefore affect other of the planner's goals. Instead, this sort of strategy is accommodated by the more general conflict resolution plans of Select-New-Plan described below.

3. If the conflict is base on an adverse interaction between subgoals that have arisen in the course of developing plans for some goals, then the conflict might be resolvable by Reschedule-Tasks.

4. Another quite general plan is Select-New-Plan, which is applicable to any goal conflict in which the goal states themselves are not mutually exclusive. The planner simply tries to find another plan for one or both of the goals that avoids the conditions causing the conflict. It should be pointed out that when this meta-plan is attempted, the old plan and conflict should be saved. It may be that the new plan selected avoids this conflict but has some other desirable consequences. The planner may then have to compare this plan to previous ones to make a reasonable decision.

4.2.6 Resolve-Circularity

Resolve-Circularity means dealing with a subgoal that is identical to some ancestral goal. This meta-goal originates from the "Avoid Impossible Goals" meta-theme, as circular subgoals are ordinarily either unachievable, or the plan for the subgoal should be directly applicable to the parent goal. The meta-plans that are applicable here are goal

modification and plan modification: Either change try a new plan that does not involve the circular subgoal, or modify the goal so that it is no longer identical to a goal for which it is instrumental.

The latter plan is generally useful when the circular subgoals stem from the normal plans for their respective goals. For example, suppose one's goal were to be a flasher, for which the normal plan is involves having a raincoat. Suppose further that the planner doesn't own one, and that it's raining outside. If the normal plan proposed involves having a raincoat, it need not be abandoned altogether because of the circularity. Rather, it can be modified to "have umbrella", or "have old newspaper", neither of which would be appropriate for the superior goal. The point here is that the two goals serve quite different functions; it therefore may be possible to satisfy one by a modification that is not applicable to the other.

4.2.7 Resolve-Need

Resolve-Need occurs when a goal that is too difficult arises. It is almost the same situation as Resolve-Circularity, as the only meta-plans applicable are plan and goal modification. Here, their function is to ascertain why the goal is needed, and if possible, to come up with a modified plan that does not involve it or with a modified goal that does not cause the same problem.

4.2.8 Form-Alliance

Another tactic that can be tried if a goal is deemed too difficult is to try to couple up with another planner and use the combined resources to execute a plan that neither could do effectively alone. The plans applicable here are all the ways of encouraging someone to join in an alliance.

5.0 GOAL OVERLAP

In this section, we discuss the goal relationship of goal overlap in some detail. Goal overlap refers to those internal goal interactions in which the fulfillment of several goals simultaneously is easier than the combined fulfillment of each goal individually. For example, consider the following stories:

(4) John needed some wood finisher. While he was at the hardware store, he picked up some sandpaper.

(5) John wanted to get rid of his old car. Then he heard that Bill was in the market for a 57 Chevy, and would pay considerably more than the car was worth.

(6) John thought he could use some exercise. He also felt like he needed some fresh air, so he decided to go jogging in the park.

In each of these stories, a character has two goals that stand in a favorable relationship to one another. In story (4), John's two goals are to get some wood finisher and to get some sandpaper. Both goals are amenable to the same plan, namely, buying the items. This plan requires the planner to be at a place that sells the desired item. Since this

precondition is the same in both cases, the planner can fulfill the precondition once and then execute both plans simultaneously.

Story (5) also refers to two goals, getting rid of an old car and wanting to possess money. The second goal must be inferred by the reader. Once it was been inferred, a single plan can be seen to have the effect of fulfilling both John's goals. Likewise, in story (6), John's goals of getting some exercise and of getting some fresh air are both addressed at no additional cost to one another by virtue of a common plan.

The goals in each of the stories above have the following property: It is easier to fulfill the goals when they are considered together than when each goal is attended to independently from the other. Thus in story (4), pursuing both goals independently may lead to two trips to the hardware store, but together, only one trip is required. John might only get rid of his vehicle and not profit from it if he refrains from considering his goal of having money at the same time in story (5). In story (6), John may have to make two trips, and certainly spend more time, if he doesn't realize that a single plan is applicable to both his goals.

A situation in which the pursuit of several goals simultaneously is more advantageous than their independent pursuit is called goal overlap. In story understanding, it is important to recognize overlapping goals because this situation strongly influences the plans a character will choose. For example, suppose John had the goals mentioned in story (4), but that he made two trips to the hardware store. Since these goals overlap so as to eliminate the need for two separate trips, a reader

would need to infer an explanation for John's behavior. For example, the reader might conjecture that John is simple minded, or that he isn't getting along with his wife and is looking for excuses to get out of the house, that he is procrastinating because he is intimidated by the task he has to perform, or that he had a lapse of memory. Ignoring the goal overlap gives evidence to these theories, and together with other indications may cause the reader to infer one of them.

Since goal overlap influences the process of plan selection, a planning mechanism must be able to detect overlapping goals in order to plan effectively in multigoal situations. For example, we would be unwilling to accept a plan of going to a store twice when only one trip is necessary, or of giving away an unwanted item when it could be sold, or of wasting time performing two activities when only one is required. This would violated the "Don't Waste Resources" meta-theme. A powerful planning mechanism must be able to detect overlapping goals when they are present, and of taking this information into account in choosing a plan of action.

5.1 Kinds Of Goal Overlap

Goal overlap may be categorized further into a number of cases. Each case is characterized by a different kind of structural relationship between the goals or their plans, and is associated with its own way of reducing the total expenditure of resources and effort. First, there is a primary division into two main classes, which reflect whether the overlap is due to the nature of the goals themselves or to the plans one might use to achieve these goals.

Kinds of Goal Overlap

1. Mutual Inclusion - The goals overlap by virtue of inherent relationships between the states constituting their realization.

2. Piggybacking - The goals themselves have no inherent overlap, but some plan is applicable to more than one goal at once.

Each category is now further refined:

5.2 Mutual Inclusion

Mutual inclusion means that a planner has the same goal for more than one reason. Actually, the goals need not be literally identical, but may be in one of a number of relationships. Consider the following examples:

(7) John thought killing animals was morally wrong. He also thought that eating vegetables made one healthy.

(8) John had to stay home because he expected a visitor. He also had to stay in his study because he was trying to write a paper.

These example contain two different kinds goal relatedness, identity and entailment. For example, in story (7), John has the goal of eating vegetables for two independent reasons. Thus the identical goal arises twice independently, producing a goal overlap situation. Alternatively, one goal may be a specific instance of a more general goal. Story (8) is an instance of this case, as being in one's study is

a special case of being in one's home via logical entailment.

To determine if two goals are identical, they merely need to be matched against one another. However, determining if one goal state implies the other requires some specific knowledge. For example, the fact that "A is inside B implies that A is inside C if B is part of C" is needed to determine the relatedness of the goals in (8) above (Obviously, logical entailments could get arbitrarily complex. However, the everyday situations dealt with here seem to require only the straightforward application of heuristics such as the one just mentioned).

Each type of relatedness also has its own particular consequences for a planner or understander:

1. Identity - If the goals are in fact identical, then one simply follows a plan for the goal.

2. Entailment - If one goal is implied by the other, then the implying goal is pursued. That is, if John needs to stay home and stay in his study, he should pursue the goal of staying in his study.

These heuristics are meant to be useful both to a story understander, to interpret the behavior of a character, and to a planner, to determine its own behavior. For example, an understander that inferred the presence of mutually inclusive goals would expect the character with such goals to act in accordance with the heuristic above, and if he did not, would try to find some additional factor to explain

the character's behavior. Alternatively, a planner who found itself with such goals would use this heuristic to generate reasonably efficient behavior.

The further refinements in the category of mutually inclusive goal overlap come from the way in which the overlapping goals may come about. Recall that a goal may arise either because it is generated by a theme, or because it is instrumental to another goal. Of particular interest is the case in which the overlapping goals are instrumental to other goals. This situation is called limited subsumption to emphasis its relationship to goal subsumption.

5.2.1 Limited Subsumption

It is often the case that the establishment of a single state is required for a multiplicity of goals. In the case where these goals are recurring instances of a single goal, the situation is termed goal subsumption, and is discussed in length in the last section.

However, it may be that the establishment of a single state is instrumental to a set of goals that is of a definite number, unlike the unlimited recurrences involved in goal subsumption. In these situations, the kind of reasoning that is required of a planner or of a natural language understander is of a very different character. For example, since the goals themselves may be different and require different plans, the situations tend to be more dynamic than goal subsumption, and therefore require somewhat more elaborate reasoning. Thus while this form of goal overlap is behaviorally distinct from

subsumption, we stress the structural similarities by terming it limited subsumption.

For example, consider the following stories:

(9) John was going to go camping for a week. He went to the supermarket and bought a week's worth of groceries.

(10) Johnny wanted a toy and a candy bar. He asked his father for some money.

(11) John had to go to a conference in Pittsburgh. While he was there, he decided to call up an old girlfriend.

Example (9) is closest to a true case of goal subsumption. Here John anticipates a number of recurring satisfy hunger goals. Since he one will require that he buy food, and this requires that he be at a store, he fulfills this common precondition for all of them at once. This example is somewhat different from goal subsumption in that a goal subsumption state usually satisfies an indefinite number of repetitions of the same goal, whereas in limited subsumption the actual state brought about is a function of the number of goals anticipated. In this example, John buys enough groceries to cover one week's worth of hunger goals.

Johnny's goals in example (10) are a yet more limited form of subsumption. Here there are only two overlapping goals, both of which the planner is attempting to achieve by asking his father for money. The reader of this story uses this knowledge to infer that Johnny probably asked his father for enough money for both of his goals.

Story (11) is even further removed from a case of goal subsumption because different plans are involved in each goal, and their interaction seems accidental, not principled. That is, John had to be in Pittsburgh in order to attend a conference, and being in Pittsburgh is also instrumental to seeing someone who lives there. The two goals overlap on this precondition, although the plans of attending a conference and seeing an old girlfriend are themselves not similar. Rather, the overlap appears fortuitous, and possibly even novel to some readers.

Limited subsumption occurs, then, in situations in which

1. A character has several (but a definite number of) goals
2. The normal plans for these goals require related preconditions

When limited subsumption occurs, a planner's strategy is apt to be the following:

1. Fulfill the common precondition,
2. Execute one of the plans
3. Execute the remaining plans before taking some action that undoes the common precondition.

In the case in which one precondition makes it easier to fulfill another, then the same strategy is followed except that one of the plans may be suspended momentarily while another is executed, and continued after the other plan is completed.

These heuristics are actually somewhat more general than has been claimed, since they are applicable to cases in which some, but not all of the overlapping goal states are instrumental to other goals. That is, a goal stemming directly from a theme may overlap with a goal that is instrumental to another goal. The same heuristics still apply. The general principle embodied here is that a state should be maintained until it is no longer instrumental to any unfulfilled goal.

5.3 Piggybacking

Occasionally, a fortuitous situation arises in which the execution of a single action simultaneously fulfills a number of distinct goals. The usual form this situation takes is that a single action achieves one of the goals directly, and the other by what is normally considered a side effect. For example, consider the following stories:

(12) John had a crack on his wall. He decided to cover it with his favorite poster.

(13) John decided he should be more physically fit. He was also something of a masochist, so he decided to take up jogging.

(14) John wanted to go see the new disaster movie. He also wanted to see Mary, so he called her up and asked her if she'd go see the movie with him.

In story (12), John's goals are to put up a poster that he likes to look at, and to hide a crack on his wall. Since a side effect of putting up a poster is to cover the area behind it, one plan can be executed that accomplishes both goals simultaneously. The goals in

story (13) are to get in shape and to experience a little discomfort. Again, both can be accomplished through the execution of a single plan. In story (14), one plan also accomplishes two goals. Here the goals are a little more equal, however. The plan of going on a date is normally associated with several goals, including enjoying someone's company and enjoying some activity. Thus the selection of this plan is a fairly stereotyped procedure for satisfying several goals at once.

Since a plan that simultaneously fulfills a number of goals is generally a normal plan for one of these goals and just happens to fulfill another goal in a particular set of circumstances, we refer to these relationships as goal piggybacking. These situations are said to occur when

1. One character has several goals
2. The normal plan for one goal has a consequence that fulfills the other goals

We include in this definition those cases in which the normal plan for one goal also happens to be the normal plan for the others. This is the case in example (14) above.

When a goal piggybacking situation arises, it is expected that the planner execute that single plan that fulfills both goals. This may require the planner to execute this plan in a particular manner. For example, in story (12), the plan piggybacks both goals only if the poster is put on the wall over the crack. This requires that the understander or planner realize that the goal of covering the crack is a

specific instance of the state that results as a side effect of putting up the poster. This relationship is discussed further in the next section.

Goal piggybacking is related to goal subsumption in that many subsumption states address a number of goals at once. For example, social relationships like marriage usually subsume social, emotional, pragmatic and financial goals simultaneously. Thus establishing such a subsumption state is a plan that piggybacks the goals of subsuming all these needs. A person might marry primarily for love, for example, but also hope to serve secondary goals involving beauty and wealth as well.

5.3.1 Closeness

One special case of piggybacking occurs when the execution of a plan for one goal makes it easier to fulfill another goal, rather than accomplishes that goal outright. For example, consider the following stories:

(15) John needed some instant pudding. He decided to pick some up at the supermarket he passed on the way home from work.

(16) John was shopping for a watch for himself. Then he noticed that the store was having a fabulous one cent sale, so he bought a watch as a present for Mary as well.

Both stories (15) and (16) contain pairs of goals that are related to one another only in how the plans for the goals interact. In story (15) for example, John has the goal of possessing some instant pudding, and the goal of being at home. Being at a supermarket is a precondition

for a plan for the first goal, and executing a plan for the second reduces the distance John must travel to achieve this state. Thus John plans to execute his shopping plan in the middle of his plan of driving home as a way to minimize his effort.

In story (16), John's goals are to have a watch, and to get a gift for Mary. Since the purchase of one watch vastly reduces the price of another, John decides to execute a plan to get Mary's gift at this point, minimizing the cost of that plan.

A state in which the amount of a resource needed to fulfill a goal is trivial is said to be "close" to the goal. Thus in stories (15) and (16), the execution of one plan brings about a state that is close to a state involved in the fulfillment of another goal. Being near the supermarket reduces the cost required to be at the supermarket, and buying one watch reduces the price of the second watch.

Determining whether one state makes it easier achieve another requires heuristic planning knowledge specific to the particular states involved. For example, a heuristic about location states that the nearer you are to a place, the easier it is to get there. This heuristic is needed in (15) to understand why stopping off at a store is a good idea.

Once it has been determined that a goal overlap situation based on closeness exists, the following plan scheduling heuristic applies. The planner should pursue the first plan until it effects the "close" state, then suspend that plan, pursue the other plan to completion, and then resume the suspended plan. For example, in story (15), John pursues his

plan to get home until he is near the store, then goes to the store and makes his purchase, and finally, resumes his plan of driving home. In story (16), the plan for buying the second object needs to be executed at the point where the plan for the first object is to be executed.

The idea of closeness is important in a number of other planning contexts. For example, in dealing with a number of goals at the same time, one often has to suspend execution of one plan and pursue the execution of another. Usually, it is undesirable to undo a state that is a precondition for an active plan while pursuing another plan. However, it is usually allowable to undo such a state if the resulting state is close to the one undone. Thus a planner who is waiting at a location to meet someone might cross the street to buy a paper, say, even though this violates a previously established condition, since the cost of restoring this state is normally considered to be small.

6.0 APPLICATIONS

We are currently attempting to use meta-planning in two programs. PAM, a story understanding system, uses knowledge about goal interactions to understand stories involving multiple goals. That is, PAM can detect situations like goal conflict and goal competition, and, realizing that these threaten certain meta-goals, PAM will interpret a character's subsequent behavior as a meta-plan to address the negative consequences of these interactions. As PAM has been discussed at length elsewhere, and its relation to meta-goals discussed above, we will not discuss it further here.

Meta-planning is also being used in the development of a planning program called PANDORA (Plan ANalyzer with Dynamic Organization, Revision and Application). PANDORA is a planning system developed along the lines described toward the beginning of this paper. PANDORA is given a description of a situation and determines if it has any goals it should act upon. It then creates plans for these goals. PANDORA is dynamically told about new developments, and changes it plans accordingly. PANDORA's ultimate objective is to act both as an independent planner and as a planning assistant that suggests plausible plans to try in complicated situations.

The following is typical of the kind of situational planning PANDORA is capable of working on. PANDORA is presented with a situation in which it believes it is cooking dinner for itself. PANDORA then receives a call from an old flame, who's only in town for a short while. PANDORA infers that it has the goal of meeting with this old friend, and that this goal is in conflict with the original goal of preparing dinner as they both occupy the same time slot. Realizing that the "Achieve As Many Goals As Possible" meta-theme is activated by this situation, PANDORA infers that it has a Resolve-Conflict goal. It now looks for a meta-plan for this goal. One such plan currently available to PANDORA is Plan Integration. That is, PANDORA generates a new plan that modifies the old one by inviting the friend to join it for dinner.

References

1] Barstow, David R. (1977). Automatic construction of algorithms and data structures using a knowledge base of programming rules. Standford AI Memo AIM-308.

2] Bruce, B. (1975) Generation as a Social Action. Proceedings of the Conference on Theoretcial Issues in Natural Language Processing, Cambridge, Mass.

3] Fikes, R. and Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence 2, 189-208.

4] Hayes-Roth, B. and Hayes-Roth, F. (1978). Cognitive Processes in Planning. RAND Report R-2366-ONR.

5] McDermott, Drew (1977). Flexibility and efficiency in a computer program for designing circuits. Cambridge: MIT AI Laboratory Technical Report 402.

6] McDermott, Drew (1978). Planning and Acting. In Cognitive Science vol. 2, no. 2.

7] McCarthy, J. and Hayes, P. J. (1969) Some philosophical problems from the standpoint of artificial intelligence. In Meltzer and D. Michie (eds.) Machine intelligence, vol. 4. New York: American Elsevier, pp. 463-502.

8] Newell, A., and Simon, H. A. (1972). Human Problem Solving. Englewood Cliffs, N. J.: Prentice Hall.

9] Perrault, C. R., Allen, J. F., and Cohen, P. R. (1978). Speech acts as a basis for understanding dialogue coherence. In Proceedings of the second conference on theoretical issuses in natural language processing. Champaigne-Urbana, Illinois.

10] Rich, C. and Shrobe, H. (1976). Initial report on a LISP programmer's apprentice. MIT AI Lab Technical Report 354.

11] Rieger, C. (1975). One System for Two Tasks: A Commonsense Algorithm Mmeory that Solves Problems and Comprehends Language. MIT AI Lab working paper 114.

12] Sacerdoti, E. D. (1974). Planning in a hierarchy of abstraction spaces. Artificial Intelligence, vol. 5, no. 2, pp. 115-135.

13] Sacerdoti, E. D. (1977). A Structure for Plans and Behavior. Elsevier North-Holland, Amsterdam.

14] Schank, R. C. and Abelson, R. P. (1977). Scripts, Plans, Goals, and Understanding. Lawrence Erlbaum Press, Hillsdale, N.J.

15] Sussman, G. J. (1975). A Computer Model of Skill Acquisition. American Elsevier, New York.

16] Wilensky, R. (1978). Understanding goal-based stories. Yale University Research Report #140.

17] Wilensky, R. (1980). Points: A theory of story content. Berkeley Electronic Research Laboratory Memorandum No. UCB/ERL/M80/17.