COGNITIVE SCIENCE **11**, 163–200 (1987)

# A Plan Recognition Model for Subdialogues in Conversations

DIANE J. LITMAN
*AT&T Bell Laboratories*

JAMES F. ALLEN
*Department of Computer Science*
*University of Rochester*

Previous plan-based models of dialogue understanding have been unable to account for many types of subdialogues present in naturally occurring conversations. One reason for this is that the models have not clearly differentiated between the varoius ways that an utterance can relate to a plan structure representing a topic. In this paper we present a plan-based theory that allows a wide variety of utterance-plan relationships. We introduce a set of discourse plans, each one corresponding to a particular way that an utterance can relate to a discourse topic, and distinguish such plans from the set of plans that are actually used to model the topics. By incorporating knowledge about discourse into a plan-based framework, we can account for a wide variety of subdialogues while maintaining the computational advantages of the plan-based approach.

## 1. INTRODUCTION

*Task-oriented dialogues* occur when two people work cooperatively on a task (i.e., a plan) which is performed during the dialogue. One promising approach to analyzing such dialogues has involved modeling the plans of the speakers in the task domain. The earliest work in this area involved tracking the topic of a dialogue by tracking the progress of the plans of the conversants (Grosz, 1977), as well as explicitly incorporating speech acts into a planning framework for single utterances (Allen and Perrault, 1980; Cohen & Perrault, 1979). A good example of the current status of these approaches can be found in Carberry (1983, 1986) and Sidner (1985). In general, these models work well as long as the topic follows the task structure closely, but encounter difficulty accounting for subdialogues such as clarifications, corrections, and topic change.

---

The preparation of this paper was supported in part by the National Science Foundation under Grant IST-8210564, the Office of Naval Research under Grant N00014-80-C-1097, and the Defense Advanced Research Projects Agency under Grant N00014-82-K-0193.

Correspondence and requests for reprints should be sent to Diane J. Litman, AT&T Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974.

One reason for this difficulty is that the models have not clearly differentiated between the various ways that an utterance can relate to a plan structure representing a topic. In particular, previous theories have only allowed an utterance to describe a step in a plan (e.g., Grosz, 1977) or to be a step in a plan (e.g., Allen and Perrault, 1980). As we will see, examination of some example dialogues will illustrate that there are many other ways that an utterance can relate to a plan. For example, utterances may change a plan, clarify a plan, correct a plan, and so on. This article presents a plan-based theory allowing for a wide variety of such utterance-plan relationships. The theory integrates knowledge about plans with knowledge about discourse, allowing for a wide variety of subdialogues while maintaining the computational advantages of the plan-based approach.

Consider the following three dialogue fragments, illustrating a topic change, a clarification, and a correction subdialogue, respectively. Dialogue 1 is a constructed dialogue, while Dialogue 2 is from a corpus of transcripts recorded at an information booth in the Toronto train station (Horrigan, 1977). Dialogue 2 is similar to the *information-seeking* dialogues found in Carberry (1983), dialogues in which an agent seeks information with respect to a plan which is not executed during the dialogue. Dialogue 3 is a slightly modified portion of a task-oriented scenario developed from protocols of a graphics system that displays network structures (Sidner & Bates, 1983). Unlike in the previous dialogues, the system's interaction with the user is primarily nonlinguistic, with utterances only being produced to satisfy simple conversational conventions.

| | | |
|---|---|---|
| (1) | Passenger: | I'd like to buy a ticket to New York please. |
| (2) | | How much is it? |
| (3) | Clerk: | Five dollars. |
| (4) | Passenger: | Here's a ten. <hands Clerk ten dollars> |
| (5) | | *By the way, I'd like to buy a newspaper.* |
| (6) | | *Is there a newsstand around here?* |
| (7) | Clerk: | *There's one down the corridor there.* |
| (8) | Passenger: | *Thanks.* |
| (9) | | Now, exactly where do I catch the train? |

Dialogue (1): A Topic Change Subdialogue

| | | |
|---|---|---|
| (10) | Passenger: | The eight-fifty to Montreal? |
| (11) | Clerk: | Eight fifty to Montreal. Gate seven. |
| (12) | Passenger: | *Where is it?* |
| (13) | Clerk: | *Down this way to the left. Second one on the left.* |
| (14) | Passenger: | OK. Thank you. |

Dialogue (2): A Clarification Subdialogue

| | | |
|---|---|---|
| (15) | User: | Show me the generic concept called "employee." |
| (16) | System: | OK. <System displays network> |
| (17) | User: | *No, can you move the concept up?* |
| (18) | System: | *Yes. <System redisplays network>* |

(19) User: OK, now make an individual employee concept whose first name is "Sam" and whose last name is "Jones."

Dialogue (3): A Correction Subdialogue

The plans underlying Dialogues 1 and 2 are sketched in Figures 1 and 2. Each figure shows the decomposition of plans into subplans, with an implicit temporal ordering from left to right. Thus, TAKE-TRIP consists of a BUY-TICKET plan (which itself consists of three subplans), followed by a GOTO-TRAIN plan, followed by a GETON-TRAIN plan. Plans will be formally defined later in the paper. For now, let us use these plans to examine each utterance-plan relationship in the dialogues.

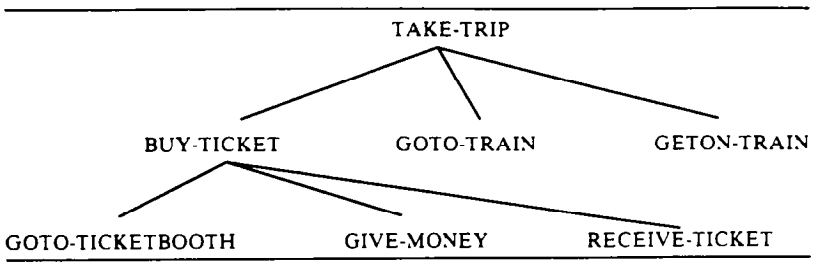


Figure 1. Sketch of plan to take a trip.

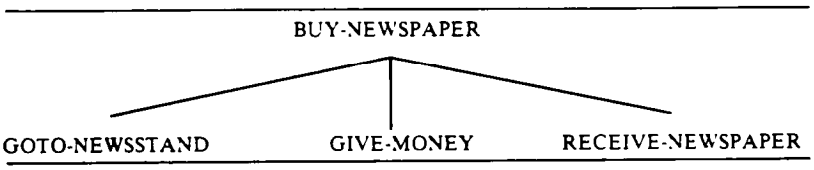


Figure 2. Sketch of plan to buy a newspaper.

Except for the temporary plan change of utterances (5)–(8), Dialogue 1 contains utterance-plan relationships that were implicitly allowed in previous formulations:

(1) Passenger: "I'd like to buy a ticket to New York please."

This utterance explicitly identifies the speaker's plan as BUY-TICKET (and implicitly as TAKE-TRIP). In essence, Passenger has introduced BUY-TICKET (and TAKE-TRIP) as the topic of conversation.

(2) Passenger: "How much is it?"

This is a query about one of the parameters in the plan introduced by utterance (1). In particular, Passenger is asking for the price of a ticket, which Passenger needs to know to perform the GIVE-MONEY step of BUY-TICKET. At another level of analysis, Passenger has requested that Clerk

perform some action (i.e., answering a question) that is related to the BUY-TICKET plan.

(3) Clerk: "Five dollars."

This utterance relates to the BUY-TICKET plan in much the same way as utterance (2). Another relevant analysis, however, is that Clerk did exactly what was expected given the question of utterance (2).

(4) Passenger: "Here's a ten."

Passenger is actually executing GIVE-MONEY at the time of this utterance. Thus, the utterance describes or accompanies a step in the BUY-TICKET plan, but is not an essential part of the plan itself.

(5) Passenger: "By the way, I'd like to buy a newspaper."

Here Passenger uses an utterance that has no direct relation to the BUY-TICKET or TAKE-TRIP plans, and thus changes the topic of conversation. In particular, utterance (5) introduces a new speaker plan BUY-NEWSPAPER, in the same way that utterance (1) introduced the plan BUY-TICKET. Discussion of the previous plan, TAKE-TRIP, is temporarily suspended.

(6) Passenger: "Is there a newsstand around here?

This can be analyzed in the same way as utterance (2), now with respect to the new plan BUY-NEWSPAPER. The utterance is a question about one of the parameters (i.e., the newsstand location) in the GOTO-NEWSSTAND step of BUY-NEWSPAPER.

(7) Clerk: "There's one down the corridor there."

This is similar to the analysis of utterance (3), and is the expected continuation given utterance (6).

(8) Passenger: "Thanks."

This acknowledges the success of the previous interaction and has only indirect relevance to the BUY-NEWSPAPER plan.

(9) Passenger: "Now, exactly where do I catch the train?"

This utterance has no relation to the BUY-NEWSPAPER plan and instead is a question about the location parameter of the GOTO-TRAIN step of TAKE-TRIP. Utterance (9) thus ends the discussion of BUY-NEWSPAPER and resumes the discussion of TAKE-TRIP.

To summarize, Dialogue 1 contains utterances that accompany the execution of a plan (e.g., (4)), that introduce a plan as a topic of discussion (e.g., (1) and (5)), that talk about some aspect of a plan (e.g., (2), (3), (6), and (7)), that resume discussion of a previous plan (e.g., (9)), and that only indirectly relate to a plan (8).

The remaining two dialogues introduce two other common forms of subdialogues. Dialogue (2) starts with a request for information about a parameter in a TAKE-TRIP plan (i.e., a request for information about the train being taken). Note that here the plan introduction is implicit in the question, rather than explicit as in Dialogue (1). The clerk provides the expected response (11), then the passenger continues with a question about this response. Note that unlike similar situations in the previous dialogue, the passenger's question is concerned with the answer provided in (11) rather than with the TAKE-TRIP plan directly. We thus consider (12) to be the start of a clarification subdialogue. Utterance (13) provides the expected response to question (12), and the dialogue is terminated by (14).

In Dialogue (3) we see one more form of subdialogue. The dialogue starts with a request (15), which implicitly introduces the plan forming the topic of conversation. The system responds as expected (16), but rather than continuing with the plan, the user instead starts to talk about correcting the plan (17). This is because the location of the displayed network does not leave enough space for the user to continue. Only after this correction is performed (18) can the user resume the original plan in the now corrected environment (19).

In this paper we present a theory that differentiates between the different ways that an utterance can relate to a plan (or plans) representing the topics of a conversation. Our theory is specified using the same plan framework previously used for representing topics. We define a set of *discourse plans,* each one corresponding to a particular way that an utterance can relate to the current discourse topic, and distinguish these plans from the set of plans that are actually used to model the topics. More specifically, *domain plans* are defined to be the speaker plans that form the topic of conversation (e.g., take a trip), while *discourse plans* are defined to be the speaker plans that relate utterances to these domain plans (e.g., introduce take a trip). This plan-based formulation of utterance-plan relationships has several important advantages. First, plans are a relatively well-defined method of representation. More importantly, however, the techniques of plan recognition can be used to identify appropriate discourse plans given actual utterances. Finally, note that in clarification subdialogues the topic of conversation corresponds to the previous discourse interaction itself. That is, plans that usually are considered discourse plans may themselves become the topic of conversation, and thus temporarily be viewed as domain plans! Since both the topic and the discourse interaction are represented as plans, we can use the same techniques to handle clarifications to an arbitrary depth of nesting (in principle).

The next section details our new model of plan recognition. We outline a simple theory of plans that is adequate for the present purposes of this paper, show how discourse and domain plans are defined in this framework. then present a plan recognition technique for recognizing discourse and

domain plans from actual utterances. We illustrate the model with a detailed analysis of the processing of Dialogues (2) and (3) above.

## 2. THE PLAN RECOGNITION MODEL

### 2.1 Overview

At any stage of a dialogue the current discourse context is represented by a stack of plans. Since at the beginning of a dialogue there are no recognized plans, the initial discourse context is an empty stack. In the simplest case after a dialogue has started, the stack consists of only two plans. The bottom plan is a domain plan representing the current topic of conversation, while the top plan is a discourse plan representing the last interaction relating to this topic. In more complex cases such as in clarification subdialogues, additional discourse plans may be on the stack, with each discourse plan referring to the plan directly below it as its topic. Finally, during a topic change several domain plans may be on the stack, with the uppermost one being the current topic and the lower ones representing suspended topics that may later be resumed.

Each new utterance must be classified as part of a new discourse plan, which itself must then be related to an appropriate domain plan on the plan stack. In the simplest case, an utterance relates to the single domain plan on the stack. The completed discourse plan representing the previous interaction is popped,and the new discourse plan pushed in its place. In other cases, both old domain and discourse plans can remain on the stack, and additional domain and discourse plans pushed on top of them.

The set of discourse plans is relatively small and will be itemized in a later section. For the moment, consider two simple examples. INTRODUCE-PLAN is a discourse plan that causes introduction of a new topic. This is achieved by popping the previous discourse plan from the stack (leaving the previous domain plan on top), then pushing a new domain plan representing the new topic, followed by the INTRODUCE-PLAN representing the topic introduction interaction itself. In contrast, the discourse plan CONTINUE-PLAN simply continues the top plan on the stack, or a lower plan if the top one is completed. In this case, any completed plans are popped off the stack before the CONTINUE-PLAN is pushed on top.

The plan recognizer attempts to classify each utterance in terms of a discourse plan by using a predetermined preference ordering of these plans. CONTINUE-PLAN is the first discourse plan considered. In effect, this means that if an utterance can be viewed as an expected continuation of the current topic, then it will be recognized as such, even though it might possibly have some other interpretation. INTRODUCE-PLAN, on the other hand, is the last discourse plan to be considered. This means that all other interpretations will be tried before the utterance is considered as a change of

topic. As we shall later discuss, this preference ordering may be changed by the presence of so-called "clue words" in the dialogue—words or phrases often conveying discourse information. For example, if an utterance is preceded by the phrase "By the way," then the INTRODUCE-PLAN interpretation would be tried before the CONTINUE-PLAN interpretation.

### 2.2 The Plan Stack

During a dialogue, a stack of executing, suspended, and completed plans is built. Several models of discourse (e.g., Polanyi & Scha, 1983; Reichman-Adar, 1984) have argued that topic structure follows a stack-like discipline. Each discourse plan on the stack refers to the plan below it, with the domain-dependent task plan at the bottom and the currently executing interaction at the top. In earlier plan-based systems, execution of one domain plan (which could itself be modeled with a stack (Grosz, 1977)) constituted the dialogue processing.

In this paper, the stack of plans will always represent what the system believes is the state of the joint plan. Because both agents may construct and execute these plans, however, at times it will seem that the stack is not truly a stack. This occurs when the user acts and the system has to recognize what sequence of planning and execution steps the user did. For example, if the user popped the top plan, and executed a step in what is now the user's top plan, the system would recognize this as executing part of a plan in the second from the top plan. This anomaly is quickly resolved as the system can then pop its stack to bring the two agents' views back into synchronization. Thus, once the plan recognition process is completed, the observed action is always in the plan that is on the top of the stack.

To rephrase this, plans are added and deleted according to the stack discipline. The plan recognizer, however, is allowed to inspect the entire stack in order to recognize that the user has popped the stack before the user executed the recognized action. Even when the system believes the top plan has completed successfully, it cannot be popped before some acknowledgement from the user, thus allowing for a clarification of the complete plan. The acknowledgement could be explicit, but most often is implicit in that the user acts in such a way that the system recognizes that it must pop the top plan.

The stack can thus be viewed as having a mixture of both suspended (partially executed) and completed plans. At the top of the stack there is a set of plans (possibly null) that the recognizer believes has been executed and completed. As just discussed, these plans cannot be eliminated as a possible topic until the user acknowledges their successful completion. Below these plans is at least one currently suspended plan. Each suspended plan will be resumed when the one above it is popped. In the case when there are no completed plans, the top suspended plan is also believed to be executing. Thus,

the top of the stack is either an executing or just-executed plan. The rest of the stack may contain other suspended or completed plans. If the stack is empty, a task will either be introduced or has just been concluded. The actual pushing and popping of plans will be discussed in the section on the plan recognition algorithm.

As an example, a clarification is modeled by a plan structure that refers to the plan that is the topic of the clarification. When a clarification plan is recognized, it is pushed onto the stack. The previous top, the plan being clarified, is temporarily suspended. Such a stack is shown in Figure 3. When the clarification is complete and its success acknowledged, the stack is popped and resumption of the previous plan is recognized.

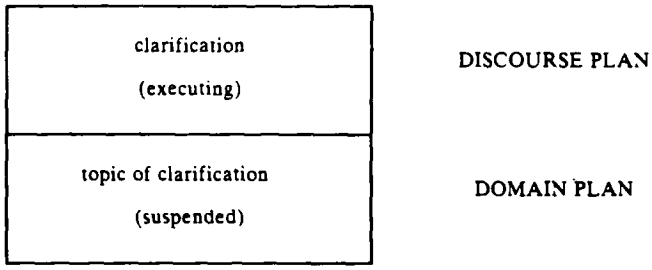


Figure 3. A clarification plan stack.

A stack metaphor obviously is an idealization for naturally-occurring conversations. For example, interrupted topics are not always returned to. In terms of the plan recognition model, we shall see that while an interpretation that corresponds to the stack discipline is preferred to one that doesn't, if no such choice exists the non-stack-like behavior will be pursued. Our assumption that there is a single shared stack between speakers is also too strong, preventing, for example, many forms of miscommunication (Goodman, 1984).

### 2.3. Representing the Plans

Plans are actions and states connected by causality and subpart relationships. Every plan has a *header,* a parameterized action description that names the plan. The *parameters of a plan* are the parameters in the header. As usual in many models of planning (e.g., STRIPS (Fikes & Nilsson, 1971); NOAH (Sacerdoti, 1977)), plans may contain *prerequisites, effects,* and a *decomposition.* Prerequisites are conditions that need to hold (or be made to hold) before the plan can actually be applied. Effects are statements that will hold after the plan has been successfully executed. We will ignore most prerequisites and effects throughout this paper, except when needed in examples. Decompositions enable hierarchical planning. Although a plan may

be usefully thought of as a single action at the level of description of the header, the plan may in actuality be decomposed into primitive actions and other abstract action descriptions (i.e., other plans). Such decompositions may be sequences of actions, sequences of subgoals to be achieved, or a mixture of both. For the purposes of this study, we are ignoring all temporal complexities in plans; plans are simply a linear sequence of actions.

Also associated with each plan is a set of applicability conditions called *constraints.* These are similar to prerequisites, except that the planner never attempts to achieve a constraint if it is false. Thus, any action whose constraints are not satisfied in some context will not be applicable in that context.

A library of *plan schemas* will be used to represent knowledge about typical speaker plans. *Plan instantiations* are formed from such general schemas by giving values to the schema parameters. We will use the term *plan* to refer to both plan schemas and instantiations. The intended meaning should always be clear from the context. We also will generally not make a distinction between the terms *plan* and *action.*

As an example, the first plan in Figure 4 indicates a primitive action, GOTO, and its effect, while the second through fourth plans indicate nonprimitive actions. For example, the third plan summarizes a simple plan schema with header "BOARD (agent, departTrain)," with parameters "agent" and "departTrain," and with a decomposition consisting of the two steps indicated. The prerequisites, effects, and constraints are not shown. Other plans needed in this domain include plans to select trains, plans to buy tickets, and so on.

| | |
|---|---|
| HEADER: | GOTO(agent, location, time) |
| EFFECT: | AT(agent, location, time) |
| HEADER: | MEET(agent, arriveTrain) |
| DECOMPOSITION: | GOTO(agent, gate(arriveTrain), time(arriveTrain)) |
| HEADER: | BOARD(agent, departTrain) |
| DECOMPOSITION: | GOTO(agent, gate(departTrain), time(departTrain)<br>GETON(agent, departTrain) |
| HEADER: | TAKE-TRAIN-TRIP(agent, departTrain, destination) |
| DECOMPOSITION: | SELECT-TRAIN(agent, departTrain, departTrainSet)<br>BUY-TICKET(agent, clerk, ticket)<br>BOARD(agent, departTrain) |
| CONSTRAINTS: | EQUAL(destination, station(departTrain))<br>EQUAL(destination, station(departTrainSet))<br>EQUAL(departTrain, object(ticket)) |

Figure 4. Plan schemas for the train domain.

Throughout this paper we assume that formulas are expressed in a typed logic that will be reflected in the naming of variables. Thus, in the above formulas, *agent* is restricted to entities capable of agency, *departTrain* is restricted to trains, and so on. As we shall see later, types are organized into type hierarchies as commonly found in semantic network formalisms.

Plan schemas can be used for both plan generation and plan recognition. For example, a planner would use these schemas in the same way it would have used STRIPS action descriptions (Fikes & Nilsson, 1971), e.g., to generate sequences of instantiated schemas to achieve some goal. Once generated, the complex plan is executed much as one would run a program. A plan recognizer, on the other hand, will use the plan schemas to recognize the plan instantiation which produced an executed action. In particular, the recognizer will be concerned with recognizing plans from actions executed as part of a dialogue.

### 2.4. Discourse Plans

In addition to domain-dependent plans, we introduce into the plan-based framework some knowledge about the conversational process itself. We do this using a small set of *discourse plans,* domain-independent plans that refer to other plans. Discourse plans represent the conversational relationships between utterances and domain plans. For example, there are discourse plans that allow utterances to introduce discussion of domain plans, to continue discussion of domain plans, and so on. Domain plans thus model the contents of a topic, while discourse plans model the actual manipulations of a topic. As we will see, a discourse plan will be recognized from every utterance, and recognition of an associated domain plan recursively generated by this recognition.

Except for the fact that discourse plans have other plans as parameters (and are thus technically meta-plans), discourse plans are identical in structure to domain plans. Before we can define them fully, however, we need to introduce some language for referring to other plans. Developing a fully adequate formal theory of this is a large research effort in its own right and cannot be addressed here. Our approach so far is meant to be suggestive of what is needed, and is specific enough for our preliminary theory and implementation.

To talk about the structure of plans, we extend our ontology to include plans as objects. We then introduce several predicates that concern plans:

> PARAMETER (P, plan): asserts that the term P is a parameter of the action description in the header of the specified plan. More formally, *P* is an instantiation of one of the parameters of the plan schema that *plan* itself is an instantiation of. Given the definition of BOARD in Figure 4, PARAMETER (S, BOARD (S, TR1)) and PARAMETER (TR1, BOARD (S, TR1)) are true.

STEP (subplan, plan): asserts that *subplan* is part of the decomposition of *plan*. More formally, *subplan* is an instantiation of one part of the decomposition of the plan schema that *plan* itself is an instantiation of. Given the definition of BOARD in Figure 4, STEP (GOTO (S, gate (TR1), time (TR1)), BOARD (S, TR1)) and STEP (GETON (S, TR1), BOARD (S, TR1)) are true.

Plans are not the only objects whose structure we need to examine. In addition, we will need to refer to parameters of actions and propositions as well. Thus, we will be working in a logic admitting plans, actions, and propositions as objects. The PARAMETER predicate will be used to make assertions about the structure of all these types of objects.

Our set of discourse plans divides roughly into three small classes: the plans that effectively continue a dialogue as expected; the plans that involve clarification and correction of the previous dialogue; and the plans that introduce new plans and change the topic of a dialogue. These classes of plans can be summarized informally as follows:

**The Continue Class:**

CONTINUE-PLAN: This simply involves the agent executing the next action in the top-most non-completed plan on the stack. As such, the agent does precisely what was expected in the given situation. A common example of this is in answering a question, as in utterances (3), (7), (11), (13), and (18).

TRACK-PLAN: This involves talking about the execution of the next action in the top-most non-completed plan on the stack, in cases where the action is non-linguistic. Examples of this include utterance (4), which describes the handing of the money to the clerk, as well as various acknowledgements such as utterance (16).

**The Clarification Class:**

IDENTIFY-PARAMETER: This involves providing a suitable description for a term instantiating a parameter of an action such that the hearer is then able to execute the action. For example, to execute the action GIVE-MONEY (P, AMT), P must know some description of AMT stated in terms of dollars, say VALUE-IN-DOLLARS (AMT, 5). Examples of this discourse act include utterances (3), (7), (11), and (13).

CORRECT-PLAN: This involves specifying correction of a plan in order to ensure its success after unexpected events at runtime. Utterance (17) provides an example.

**The Topic Shift Class:**

INTRODUCE-PLAN: This introduces a new plan for discussion that is not part of the previous topic of conversation. Explicit examples are utterances (1) and (5), while Dialogues (2) and (3) each contain a single plan introduced implicitly by utterances (10) and (15), respectively. For example, utterance (15) is a request that the system perform an action. Since actions are part of plans, the plan is implicitly introduced by the utterance.

MODIFY-PLAN: This introduces a new topic by performing some modification on the previous topic. There is no instance of this in the dialogues presented in this paper, but an example might be to replace utterances (8)-(9) with "How about a restaurant?"

While other discourse plans in each of these classes could be developed, recognition of the small set shown will be sufficient to understand an interesting class of dialogues in several domains.

Let us now consider the formal definitions of some of these discourse plans, for example, the definition of INTRODUCE-PLAN as shown in Figure 5. INTRODUCE-PLAN takes a plan of the speaker that involves the hearer and introduces it into the conversation. As discussed above, one way that a speaker may do this is by asking the hearer to perform some action (the DECOMPOSITION shown). The definitions of linguistic actions such as REQUEST will be provided in the next section. The requested action must be a step in the plan being introduced and, furthermore, the agent of the requested action must be the hearer (the CONSTRAINTS). The effects

| | |
|---|---|
| HEADER: | INTRODUCE-PLAN(speaker, hearer action, plan) |
| DECOMPOSITION: | REQUEST(speaker, hearer, action) |
| EFFECTS: | WANT(hearer, plan)<br>NEXT(action, plan) |
| CONSTRAINTS: | STEP(action, plan)<br>AGENT(action,hearer) |

Figure 5. A discourse plan to introduce a domain plan.

of INTRODUCE-PLAN are that the hearer will adopt the plan introduced (this assumes that the hearer is cooperative), and that the plan's future execution details are noted. Other ways of introducing plans, such as the speaker explicitly telling the hearer the plan (e.g., utterance (1)), do not occur in the dialogues chosen for detailed analysis later in the paper and have been omitted from the definitions, but could easily be added.

The execution status of every plan is indicated by predicates that keep track of the part of the plan that was last executed, as well as the part of the plan to be executed next. In particular, we define:

NEXT (action, plan)—true only if *action* is the next action to be executed in *plan* (or is the next action to be executed when *plan* is resumed).

LAST (action, plan)—true only if the action was the last (i.e., most recent) part of the plan to be executed.

Among the effects of every discourse plan are assertions that update these predicates with respect to the plan referred to. For example, the second ef-

fect of INTRODUCE-PLAN marks the action requested as the action to be executed in the plan introduced. We will sometimes refer to the action that is NEXT as the part of the plan that is *in focus,* drawing an analogy with *global focus* as defined in (Grosz, 1977).

The definitions of the clarification plans are presented in Figure 6.

| | |
|---|---|
| HEADER: | CORRECT-PLAN(speaker, hearer, laststep, newstep, nextstep, plan) |
| PREREQUISITES: | WANT(hearer, plan)<br>LAST(laststep, plan) |
| DECOMPOSITION-1: | REQUEST(speaker, hearer, newstep) |
| DECOMPOSITION-2: | REQUEST(speaker, hearer, nextstep) |
| EFFECTS: | STEP(newstep, plan)<br>AFTER(laststep, newstep)<br>AFTER(newstep, nextstep)<br>NEXT(newstep, plan) |
| CONSTRAINTS: | STEP(laststep, plan)<br>STEP(nextstep, plan)<br>AFTER(laststep, nextstep)<br>AGENT(newstep, hearer)<br>~CANDO(speaker, nextstep, plan)<br>MODIFIES(newstep, laststep)<br>ENABLES(newstep, nextstep) |
| HEADER: | IDENTIFY-PARAMETER(speaker, hearer, parameter, action, plan) |
| DECOMPOSITION: | INFORMREF(speaker, hearer, term, proposition) |
| EFFECTS: | NEXT(action, plan)<br>KNOW-PARAMETER(hearer, parameter, action, plan) |
| CONSTRAINTS: | PARAMETER(parameter, action)<br>STEP(action, plan)<br>PARAMETER(parameter, proposition)<br>PARAMETER(term, proposition)<br>WANT(hearer, plan) |

Figure 6. The clarification class.

CORRECT-PLAN inserts a repair step into a pre-existing plan that would otherwise fail. More specifically, CORRECT-PLAN takes a plan having subparts that do not interface as expected during execution; the plan thus needs to be modified by adding a new goal to restore the expected interactions. The pre-existing plan has subparts *laststep* and *nextstep,* where *laststep* was supposed to enable the performance of *nextstep,* but in reality did not. Thus the plan must be corrected by adding *newstep* to the executed plan, which enables the performance of *nextstep* and thus of the rest of *plan.* As in INTRODUCE-PLAN, the plan to be corrected can be referred to by a REQUEST for an as yet to be performed step (here, either *nextstep* or *newstep*). When *nextstep* is requested, the hearer has to use the knowl-

edge that *nextstep* cannot currently be performed to infer that a correction must be added to the plan. When *newstep* is requested, the speaker explicitly provides the correction. The effects and constraints capture the plan situation described above and should be self-explanatory, with the following exceptions. MODIFIES(action2, action1) means that *action2* is a variant of *action 1,* for example the same action with different parameters or a new action achieving the still required effects. ENABLES (action1, action2) means that unsatisfied preconditions of *action2* are in the effects of *action1.*

Finally, IDENTIFY-PARAMETER describes a parameter of an action that is a step of a plan. IDENTIFY-PARAMETER may be accomplished by performing an INFORMREF act as specified in the decomposition and constraints, that is, by stating some proposition that is true of the parameter. (The definition of the speech act INFORMREF will be provided in the next section.) The description should be sufficient to allow the hearer to execute the action, all other things being equal. This effect on the hearer is summarized by the assertion

KNOW-PARAMETER (hearer, parameter, action, plan).

For example, if agent P knows how many dollars a certain ticket sells for, then

KNOW-PARAMETER (P, AMT,
GIVE-MONEY (P, AMT), BUY-TICKET (P, C, TKT))

is true. While the axiomatization of KNOW-PARAMETER is problematic, we shall only be using it in simple cases where its use is straightforward.

### 2.5. Speech Act Definitions

To model speech acts such as REQUEST (Searle, 1969), we are assuming plan-based definitions as in Allen and Perrault (1980), shown in Figure 7. However, we have modified decompositions to include the conventionalized forms of indirect speech acts (Searle, 1975). For example, the second decomposition of the REQUEST for an action is that the speaker REQUEST the hearer to INFORMIF the hearer can do the action. Such decompositions are abstractions of inference paths that could have been derived from first principles as in Allen and Perrault (1980). Although the presence of "compiled" indirect speech acts allows indirect forms to be considered easily, they are just one more option to the plan recognizer. The literal interpretation is still available and will be recognized in appropriate contexts. For example, if we set up a plan to ask about someone's knowledge (say, by an initial utterance of "I need to know where the schedule is incomplete"), then the utterance "Do you know when the Windsor train leaves?" is interpreted literally as a yes/no question because that is the interpretation explicitly expected from the analysis of the initial utterance. The speech act recognition

| | |
|---|---|
| HEADER: | REQUEST(speaker, hearer, action) |
| PREREQUISITE: | WANT(speaker, action) |
| DECOMPOSITION-1: | SURFACE-REQUEST(speaker, hearer, action) |
| DECOMPOSITION-2: | SURFACE-REQUEST(speaker, hearer, INFORMIF(hearer, speaker, CANDO(hearer, action))) |
| DECOMPOSITION-3: | SURFACE-INFORM(speaker, hearer, ~(CANDO(speaker, action))) |
| DECOMPOSITION-4: | SURFACE-INFORM(speaker, hearer, WANT(speaker, action)) |
| EFFECTS: | WANT(hearer, action)<br>KNOW(hearer, WANT(speaker, action)) |
| CONSTRAINT: | AGENT(action, hearer)[1] |
| HEADER | INFORM(speaker, hearer, propositon) |
| PREREQUISITE: | KNOW(speaker, proposition) |
| DECOMPOSITION: | SURFACE-INFORM(speaker, hearer, proposition) |
| EFFECTS: | KNOW(hearer, proposition)<br>KNOW(hearer, KNOW(speaker, proposition)) |
| HEADER: | INFORMREF(speaker, hearer, term, proposition) |
| PREREQUISITE: | KNOWREF(speaker, term, proposition) |
| DECOMPOSITION: | achieve KNOW(hearer, proposition) |
| EFFECT: | KNOWREF(hearer, term, proposition) |
| CONSTRAINT: | PARAMETER(term, proposition) |
| HEADER: | INFORMIF(speaker, hearer, proposition) |
| PREREQUISITE: | KNOWIF(speaker, proposition) |
| DECOMPOSITION-1: | achieve KNOW(hearer, proposition) |
| DECOMPOSITION-2: | achieve KNOW(hearer, ~proposition) |
| EFFECT: | KNOWIF(hearer, proposition) |

Figure 7. Speech act definitions.

method of Allen and Perrault was based on differences of belief, rather than differences of discourse context.

We have also explicitly added an effect that the hearer then believes the preconditions held if the act is done successfully. This could again be inferred from first principles, but adding it to the definition allows us to use a simple plan recognition algorithm throughout the paper.

The only other difference from Allen and Perrault (1980) is that we have added an extra parameter to the INFORMREF action and the KNOWREF

[1] Technically this is only true for the first two decompositions. When the third decomposition (and sometime the fourth) is used, the agent changes from the speaker in the decomposition to the hearer in the header. For example, this happens when "I can't reach that book" requests the tall hearer to reach it instead.

assertion. The assertion KNOWREF ($A$, $t$, $p$) means that A knows a description of term $t$, which satisfies proposition $p$.

This is simply a notational variant that is closer to the actual implementation. Thus, rather than stating the goal to know when train TR1 leaves as

"KNOWREF (A, the x: depart-time (TR1, x))"

as in Allen and Perrault (1980), we write

KNOWREF (A, ?time, EQUAL (depart-time (TR1), ?time)).

Not all such assertions involve the equality predicate. For example, the representation of the goal behind the utterance "What do you want me to do?" would be

KNOWREF (speaker, ?speaker-action, WANT (hearer, ?speaker-action)).

We can define this operator formally within a possible worlds semantics of the BELIEF operator by using "quantifying in" as done in Allen and Perrault (1980). While this analysis is not fully satisfactory, it is adequate for our present purposes.

## 2.6. Plan Recognition

*2.6.1. Overview.* The task of the plan recognizer is to recognize the discourse plan and any related domain or discourse plans underlying the production of an input utterance. That is, the plan recognizer's task is not only to infer a domain plan (as in previous systems), but also to infer an utterance's relationship with this domain plan. The plan recognizer has at its disposal a library of domain plan schemas (varying with the domain), a library of discourse and speech act plan schemas (as in Figures 5, 6, and 7), the representation of the parse of the input utterance, and the plan stack representing the current state of the dialogue. Each new utterance must be classified as part of the execution of a new discourse plan. Furthermore, the discourse plan recognized must be related to another plan either on or about to be pushed on the stack. As discussed above, in most cases the discourse plan refers to the top-most noncompleted plan on the stack. However, in topic clarifications the discourse plan may also refer to a completed plan, and in cases where a topic is unexpectedly abandoned, a discourse plan refers to a noncompleted plan lower than the top-most one. Finally, in cases of topic introduction or change the discourse plan refers to a completely novel plan that must also be recognized and added to the stack. The final output of the plan recognition process is a modified plan stack, representing the updated plan state underlying the dialogue. If the interpretation is ambiguous, a stack is created for each interpretation.

*2.6.2 Recognizing and Relating Discourse and Domain Plans.* The plan recognizer's task is to find a sequence of instantiations of plan schemas, each one containing the previous one in its decomposition,[2] that connects the surface speech act analysis produced by the parser to a discourse plan. In the simplest case, the surface speech act matches the decomposition of an underlying speech act, which itself matches the decomposition of a discourse plan.

Since every discourse plan takes another plan as an argument, recognition of any discourse plan will cause an associated domain plan to be introduced. Furthermore, the constraints relating the discourse plan to this domain plan add information about the domain plan enabling further plan recognition. For instance, the utterance "Give me a train schedule please" could be identified as a part of the discourse plan

INTRODUCE-PLAN (Passenger, Clark,
GIVE (Clerk, Passenger, Schedule), Plan).

In order to validate this interpretation, we need to satisfy the constraints of INTRODUCE-PLAN, for example

GIVE (Clerk, Passenger, Schedule) is a step of Plan,

where Plan is the plan being introduced. Thus, we need to constrain Plan so that it contains the action GIVE. This can be done by invoking the plan recognizer recursively, starting with giving a schedule as the observed action. For example, GIVE could be a step in a plan to select a train, which itself could be a step in a plan to take a train trip. Since taking a trip is a domain plan, no other plans are introduced and recursive plan recognition halts.

Once a set of plans is recognized, each is expanded top-down by adding the definitions of all steps and substeps (based on the plan libraries), until there are no unique expansions for any of the remaining substeps. Each plan is then pushed onto the stack so that the original discourse plan is on top, every discourse plan refers to the plan below it, and the domain dependent plan is on the bottom.

*2.6.3. Coherence Heuristics.* The search process described above is too general, for it does not specify the influence of the discourse context as maintained on the plan stack. For example, the plan referred to by the discourse plan often exists on the plan stack. Ignoring the presence of clue

[2] Plan chaining can also be done via effects and preconditions. Pollack (1986a, 1986b) even allows recognition of nonexistent library plans. To keep the examples simple, all plan schemas have been expressed so that chaining via decompositions is sufficient.

words for the moment, the coherence heuristics control the plan recognition process by preferring the discourse plan that corresponds to the most linguistically coherent continuation of the dialogue represented by the plan stack. In particular:

1. The recognition of continuation plans is attempted first. If the utterance can be viewed as a continuation of the top-most noncompleted plan, then it is classified as such without further consideration. The speaker is believed to be doing exactly what is expected in the given situation.
2. Clarification plans are attempted next. If the utterance cannot be viewed as a continuation, but can be viewed as a clarification of one of the plans on the stack, then it is classified as such without further consideration.
3. Topic shift plans are tried last. This interpretation is attempted only when no continuations or clarifications of plans on the stack can be found, or when the stack is empty as at the beginning of a dialogue. In either case, a domain plan is constructed rather than found on the stack.

Thus, the plan recognizer prefers an utterance interpretation that continues rather than clarifies a plan, which is considered more coherent than introducing a new plan altogether. If within one of these preferences there are still competing interpretations, the interpretation that most corresponds to the stack discipline is preferred. For example, a clarification of a recent topic is preferred to a clarification of a topic earlier in the conversation. In cases where the clarification discourse plan refers to an earlier topic, the stack must be popped down to this topic before the new discourse plan is added. Furthermore, the popped topics are then considered removed from the conversation.

While these heuristics and their ordering have not been validated with psychological experimentation, they have intuitive appeal. For example, if the first heuristic was always applicable, the discourse behavior would default to the earlier plan-based systems in which a dialogue contains no unexpected subdialogues. Preferring the second heuristic over the third corresponds to the view that without any explicit linguistic markings, topic change is always least expected. Thus, interruptions are not generally predicted but can be handled when they do occur. The heuristics also follow the principle of Occam's razor, since they are ordered as to introduce as few new plans as possible. Other models of discourse (e.g., Carberry, 1983; McKeown, 1985) use similar heuristics.

*2.6.4. Linguistic Clues.* In many models of discourse (Cohen, 1983; Grosz, 1977; Grosz & Sidner, 1986; Polanyi & Scha, 1983; Reichman-Adar, 1984; Sidner, 1985), rules of conversational coherency have been shown to govern the use of surface linguistic phenomena such as clue words and choice of reference. Thus, in addition to coherence heuristics, the plan

recognizer can also use surface linguistic phenomena to control its search. For example, correlations between clue words or phrases and the set of discourse plans can be made, e.g., "by the way" often signals either INTRODUCE-PLAN, IDENTIFY-PARAMETER, or CORRECT-PLAN, while "now" often signals CONTINUE-PLAN. If a list of such clue words is available to the parser, the parser can note the presence of clue words in the input and give them to the plan recognizer along with the utterance parse. (As in other models of discourse, we currently assume that the parser can distinguish between clue and non-clue uses of a lexical item). Clues can then be used to either reinforce or redirect the plan recognizer's coherence preference ordering given above. For example, consider recognizing a topic change that is not prefaced by a clue. Using the coherence preferences, the plan recognizer first tries to interpret the utterance as a continuation, then as a clarification, and only when these efforts fail, as a change of topic. This is because a topic change is least expected in the unmarked case. However, if a topic change is prefaced with a clue such as "by the way," a continuation is not tried first. This is because a signal for IDENTIFY-PARAMETER, CORRECT-PLAN, or INTRODUCE-PLAN is explicitly present. Clues ease the recognition of topic relationships that would otherwise be difficult (if not impossible (Cohen, 1983; Grosz & Sidner, 1986; Sidner, 1985)) to understand.

In our system, other linguistic phenomena such as elliptical utterances and referring expressions interact with the coherence heuristics in a similar manner (Litman, 1985, 1986b). In particular, the information conveyed by such phenomena takes precedence over the coherence heuristics used by the plan recognizer in the linguistically unmarked case.

*2.6.5. Plan-Based Heuristics.* Although the plan recognizer may suggest a new plan as a linguistically coherent interpretation, it may still be eliminated from further consideration by a set of heuristics based on rational planning behavior as in Allen and Perrault (1980). For example, plans that are postulated whose effects are already true are eliminated, as are plans whose constraints cannot be satisfied. There is also a heuristic that eliminates plans if setting plan parameters that are compatible with known objects equal to such objects causes a contradition. Thus, in "I'd like to go to Montreal. Where is the gate?," only plans that allow the requested gate to be equal to the gate of the train to Montreal are not eliminated. Litman (1985) contains a discussion of the technical issues generated by performing this type of nonmonotonic equality reasoning.

*2.6.6. Incremental Search.* When heuristics and linguistic constraints cannot eliminate all but one postulated plan, the chaining stops, even when a solution path has not yet been found. In other words, search terminates after branch points. For example, since BOARD and MEET plans can be

recognized from GOTO (recall Figure 4), chaining would halt with two branches if both plans were plausible. Such premature termination is typical in dialogue processing (e.g., see Carberry, 1983, and Sidner & Israel, 1981), since later utterances in a dialogue often eliminate many of the branches. Cohen et al. (1982) also argue that rational speakers would not have intended inferences in branches since such inferences could not have been drawn unambiguously. Thus, the plan recognition process halts when we have either uniquely incorporated the utterance into our expectation structure or when the search space becomes too large. In the latter case, multiple stacks are created to record each branch and the plan remains ambiguous.

## 3. EXAMPLES

### 3.1. Introduction

This section uses the framework developed in the previous section to illustrate the processing of Dialogues 2 and 3. Although the behavior to be described is fully specified by the theory, the implementation of the system corresponds only to the new model of plan recognition. All simulated computational processes have been implemented elsewhere, however. For example, the current system assumes that a highly-specialized semantic grammar (Brown & Burton, 1977) has parsed the utterances in the train domain. This allows the avoidance of some difficult parsing issues and concentration on the plan recognition model. While such a grammar has in actuality been implemented, it is currently not hooked up to the plan recognition system. Litman (1985) contains a full discussion of the implementation and associated issues of knowledge representation.

The theory at present does not concern itself with the planning or generation of natural language responses. The examples will describe what the system should do (using the actual response in the dialogue as a guide), concentrating on how the response effects the representation and analysis of the mutual plan stack.

The plan recognition system is implemented using the HORNE Reasoning System (Allen et al., 1984), a lisp-embedded horn clause theorem prover with typing and equality reasoning. Aspects of the HORNE typing system need to be elaborated on in order to understand the examples. HORNE types can have a set of distinguished function names called roles, each with an associated type. For example, the type TrainType is a subtype of the type PhysicalObjectType and has three distinguished, type-restricted roles as shown:

```
(subtype TrainType PhysicalObjectType
        (gate LocationType)
        (station CityType)
        (time TimeType))
```

In other words, role function names are just another type of object to HORNE. The notation ?fn(train1) will be used to represent a role value of train1 (where train1 is an object of type TrainType). Thus, ?fn could be any of the three role names listed above (e.g., gate). If the system later were to match ?fn(train) with a variable of type CityType, ?fn(train1) would be further restricted to the station of train1. Objects of type *proposition* and *action* are also represented in the type hierarchy and have roles defined for each argument, as is done in semantic network representations. Thus, PARAMETER (term, proposition) is defined to be true only if *term* fills a role of *proposition*.

### 3.2. Dialogue 2

This section simulates the system's processing of Dialogue 2, repeated below for convenience:

Passenger: The eight-fifty to Montreal?
Clerk: Eight-fifty to Montreal. Gate seven.
Passenger: Where is it?
Clerk: Down this way to the left. Second one on the left.
Passenger: OK. Thank you.

The example will show how the system, taking the role of the clerk, understands the passenger's utterances.

The initial state of the system consists of a library of speech acts, discourse plans, and domain plans regarding trains (i.e., the plans in Figures 4, 5, 6, and 7), and an empty stack. The following (simulated) parse of "The eight-fifty to Montreal?" is input to the system:

SURFACE-REQUEST (Person1, Clerk1,
    INFORMREF(Clerk1, Person1, ?term, EQUAL(?term, ?fn(dtrain1)))

with contraints:
    station (dtrain1) = Montreal
    time (dtrain1) = eight-fifty.

In other words, Person1 is querying the clerk about some (as yet unspecified) term, ?term, that is the value of some role of dtrain1. Dtrain1 is identified as a train from the input since trains are the only objects in the domain that are described using times and cities. Dtrain1 is restricted to a depart train (i.e., Montreal is restricted to a destination) using the preposition "to." A SURFACE-REQUEST for some sort of system INFORM is recognized from the interrogative mood of the utterance. Another possible parse of the utterance, SURFACE-REQUEST to INFORMIF, would later be eliminated since a yes/no question interpretation of this utterance is inappropriate. While our analysis is similar to that produced in Allen and Perrault (1980) for interrogative sentence fragments, Litman (1986b) contains a more general plan-

based treatment of sentence fragments (based on a parser output of a noun phrase surface form with interrogative mood).

Since the stack is empty, the plan recognizer can only construct an analysis corresponding to coherence preference (3). From the SURFACE-REQUEST, via REQUEST, chaining via decompositions produces an instantiation of the INTRODUCE-PLAN discourse-plan, as shown in Figure 8. The INFORMREF action will be referred to using the name "I1."

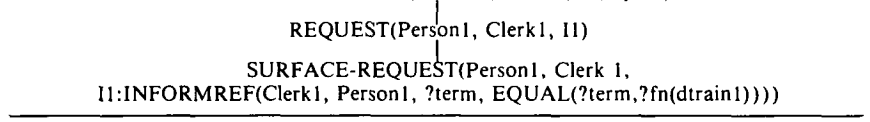


Figure 8. Chaining produces an intermediate plan recognition structure.

Before pursuing the candidate plan any further, the plan recognizer checks on the plan's reasonableness using the plan-based heuristics. From constraint satisfaction it knows that the INFORMREF must be a step in the plan being introduced. To satisfy this constraint, that is, STEP(I1,?plan), a new plan will be created and arbitrarily called PLAN2. This new state of affairs is shown in Figure 9, where the name of a plan structure appears at the top left-hand corner. In accordance with the constraint, the INFORMREF I1 is part of PLAN2. The second constraint, that is, AGENT(I1, Clerk1), is already satisfied. Finally, the recognizer verifies that the effects of the discourse plan are not already true, that is, that the clerk does not already have PLAN2 as a goal.

The recognizer continues with a recursive expansion from I1 and recognizes that PLAN2 is an IDENTIFY-PARAMETER plan, using decomposi-

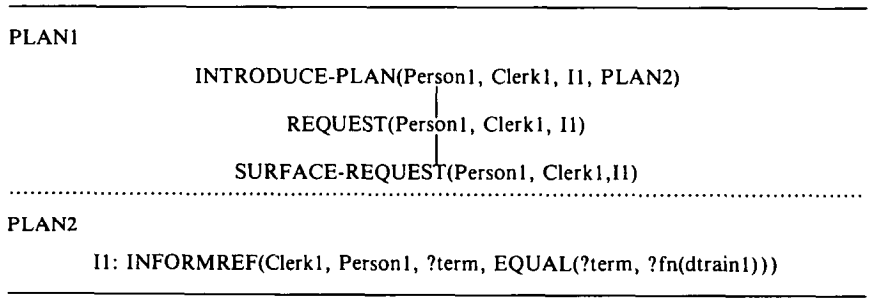


Figure 9. Constraint satisfaction initiates recognition of PLAN2.

tion chaining. In satisfying the constraints on IDENTIFY-PARAMETER (Clerk1, Person1, ?parameter, ?action, ?plan), that is,

1. PARAMETER (?parameter,?action)
2. STEP(?action,?plan)
3. PARAMETER(?parameter,EQUAL(?term,?fn(dtrain1)))
4. PARAMETER(?term,EQUAL(?term,?fn(dtrain1)))
5. WANT(Person1,?plan)

a third plan is introduced (constraint (5)) that must have a step (2) that contains a property of a train (1,3), described via the equality of the INFORMREF (4). An eligible plan is GOTO, where ?fn(dtrain1) is restricted to be the time or location of the GOTO. As a result of this, ?fn is restricted to be a time or gate role of dtrain1. Another eligible domain plan is TAKE-TRAIN-TRIP, where destination is the parameter being identified. These are the only two of the domain and discourse plan schemas satisfying all the constraints, in particular the constraint that the plan schema contains a location, city or time (i.e., a role value of dtrain1) as a parameter.[3] Both the identification of the destination in TAKE-TRAIN-TRIP and the time in GOTO are eliminated by the plan heuristic that one does not execute plans when its effects are already true. In this case, the destination and time of dtrain1 were specified in the parse and are thus already known. The plan recognition heuristics discussed above thus eliminate competing interpretations and constrain the plan to GOTO and ?fn to be the gate. Figure 10 shows the three plan structures created so far.

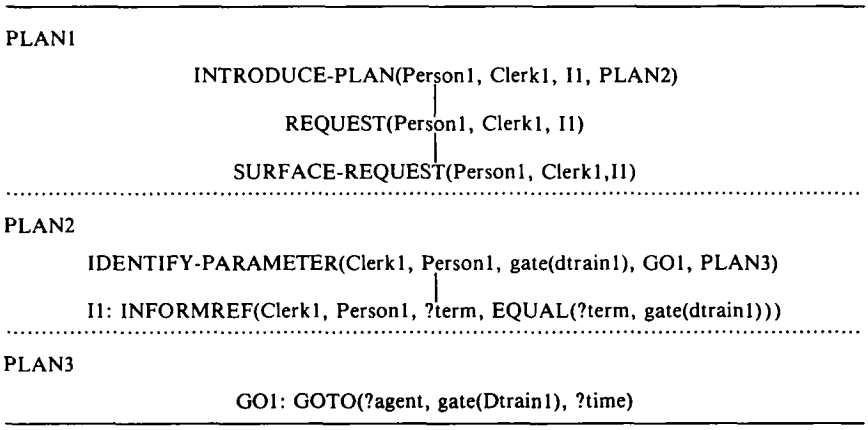


Figure 10. Constraint satisfaction creates PLAN3.

[3] The reader may be wondering why buying a ticket was not recognized as a possible plan. This was due to the simplified treatment of sentence fragments. For example, noun phrases can indicate not only questions about the noun phrase's role (as above), but also questions about roles of objects for which the noun phrase itself is a role. Thus, "The 8:50 to Montreal" could also be used to refer to a role of the train's associated ticket, for example the cost. Since BUY-TICKET would eventually be eliminated due to constraint violation (i.e., the clerk can only give information), the omission is insignificant with respect to the points being illustrated.

The plan recognition algorithm is now recursively called on PLAN2's domain plan, PLAN3. Decomposition chaining from GO1 yields both the BOARD and MEET plans. The MEET plan is eliminated due to typing constraint violation; dtrain1 is already known to be a departing train from the parse. Since the expected agent of the BOARD plan is the speaker, ?agent is heuristically set equal to Person1. Finally, chaining proceeds from BOARD to TAKE-TRAIN-TRIP, and since PLAN3 is a domain plan (i.e., no more plans are introduced), plan recognition ends.

All postulated plans are expanded top-down to include the rest of their steps, and the stack is constructed. Note that all three plans are recognized before any is placed on the stack. Furthermore, although GOTO, for example, is recognized after IDENTIFY-PARAMETER, it is put on the stack first since its execution is suspended for the IDENTIFY-PARAMETER clarification. The state of the stack after the plan recognition process just discussed is shown in Figure 11, which should be viewed as one stack with three elements: PLAN1 is at the top of the stack, PLAN2 in the middle, and PLAN3 at the bottom. The changes in the plans from the previous figure are italicized; the dotted lines indicate the information known from the top down expansions. The recognizer has constructed an entire plan stack based on the original domain-specific expectations that the speaker (although more generally it could be someone besides the speaker) will take or meet a train. In particular, the speaker has introduced a clarification of this domain plan. The introduction was executed, the clarification will be executed upon completion of the introduction, and the domain plan will be executed upon completion of the clarification.

Once the structure is recognized the executing step in each plan structure is noted, as shown in square brackets in the figures. Thus the SURFACE-REQUEST in PLAN1 is marked as the LAST (the executed) step of PLAN1, which is now completed. From the effects of INTRODUCE-PLAN and IDENTIFY-PARAMETER we mark I1 and GO1, respectively, as in focus whenever execution returns to those stacked plans.

The clerk's planning of the response is not specified in this theory. Here what the system should do will be described. The system responds with the expected continuation, namely the INFORMREF in PLAN2, popping the completed PLAN1 off the stack. To execute the INFORMREF, the system would plan A SURFACE-INFORM, which might eventually be realized as the utterance "Eight-fifty to Montreal. Gate seven." Using the assumption that this utterance is correctly recognized by the passenger, the system updates the focus in PLAN2, marking the new SURFACE-INFORM step as [LAST] and the plan as [completed]. This anticipates the passenger's correct interpretation of the utterance, since the stack is assumed to be shared between the speakers. Although all the steps of PLAN2 are completed, its success cannot be confirmed until the next utterance. Thus it is not popped off

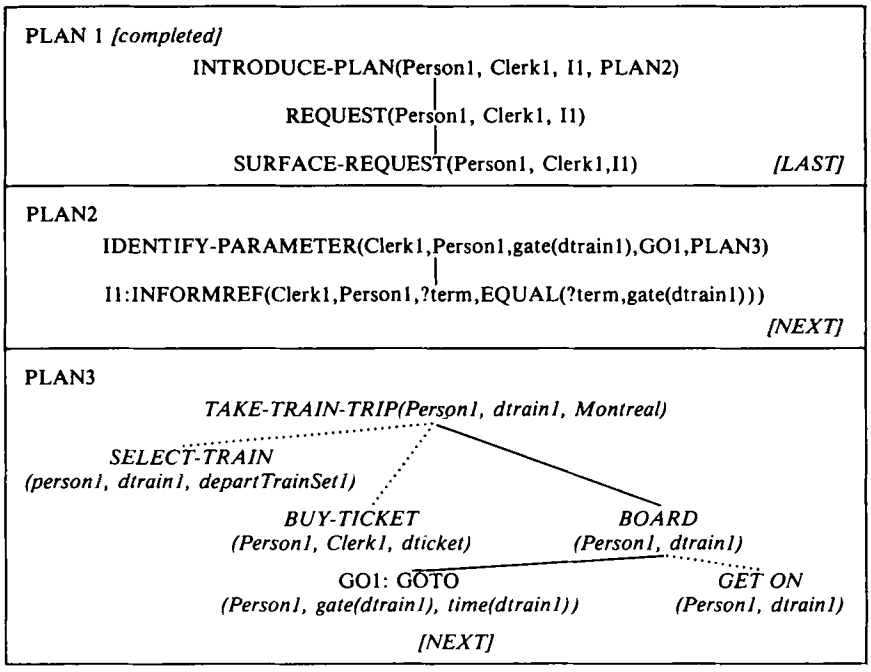


Figure 11. The plan stack after the first utterance.

the stack. This allows the possibility of a clarification plan being introduced concerning PLAN2. Thus, at the stage just before Person1 speaks again, the stack would contain PLAN3 and PLAN2, as shown in Figure 12.

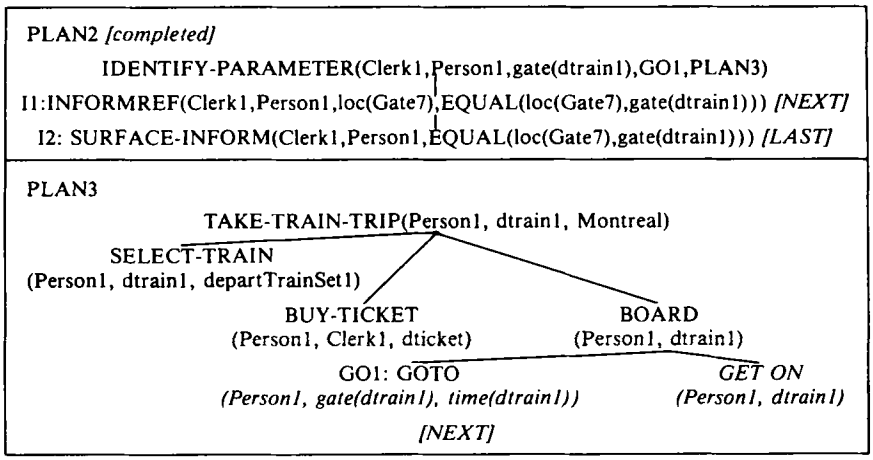


Figure 12. The plan stack after the clerk's response.

The passenger then asks "Where is it?", which would be parsed into the linguistic action:

SURFACE-REQUEST (Person1, Clerk1
INFORMREF (Clerk1,Person1,?term1,EQUAL (?term1,loc(Gate7))))

This analysis assumes the appropriate resolution of "it" to Gate7 by an immediate focus mechanism such as Sidner's (1983). This makes the example simpler. In this case the plan recognition would work even if the "it" were not resolved and left as an unknown constant, as discussed below.

Although the clerk thought the SURFACE-INFORM of PLAN2 achieved the desired passenger KNOWREF, in actuality it did not provide a description enabling the passenger to execute the GOTO of PLAN3. Instead we have a request for clarification. The plan recognizer attempts to incorporate this utterance into the plan stack based on the coherence heuristics. The first preference fails since the SURFACE-REQUEST does not match (directly or by chaining) any of the steps in PLAN3, which should have been resumed if the previous clarification was understood. This preference, then, involving popping the completed PLAN2, is not possible because the utterance cannot be seen as a step of the TAKE-TRAIN-TRIP plan. The second preference succeeds, and the utterance is recognized as part of an introduction of a new IDENTIFY-PARAMETER referring to the old one. This process is basically analogous to the process discussed in detail above, except that the plan to which the IDENTIFY-PARAMETER refers is found in the stack rather than constructed. The final results of the analysis are pushed onto the stack after popping to the appropriate plan (which is here already on top), as shown in Figure 13.

As mentioned above, if in the "it" was originally unresolved (i.e., Gate7 in I3 is unknown), "it" will be correctly resolved later as a side-effect of the plan recognition. In particular, the appropriate binding will be made during the constraint propagation process connecting PLAN5 to PLAN2.

Using the actual clerk's reply, "Down this way to the left—second one on the left" as a guide, the clerk's response is simulated as a pop of the completed PLAN4, followed by execution of the INFORMREF I3 and thus completion of PLAN5. The system is now ready to recognize the passenger's next plan.

The dialogue concludes with "OK. Thank you." which according to preference (1) should be interpreted as a pop of all completed plans, leading to a continuation of TAKE-TRAIN-TRIP. This is reinforced by the clue word interpretation of the utterance. "OK" is an example of a clue word marking (among other things) CONTINUE-PLAN, possibly after a pop off the stack. The analysis is further reinforced by "Thank you," a discourse convention also indicating some level of plan or subplan completion, as well as termination of the dialogue if not followed by further utterances. In fact,

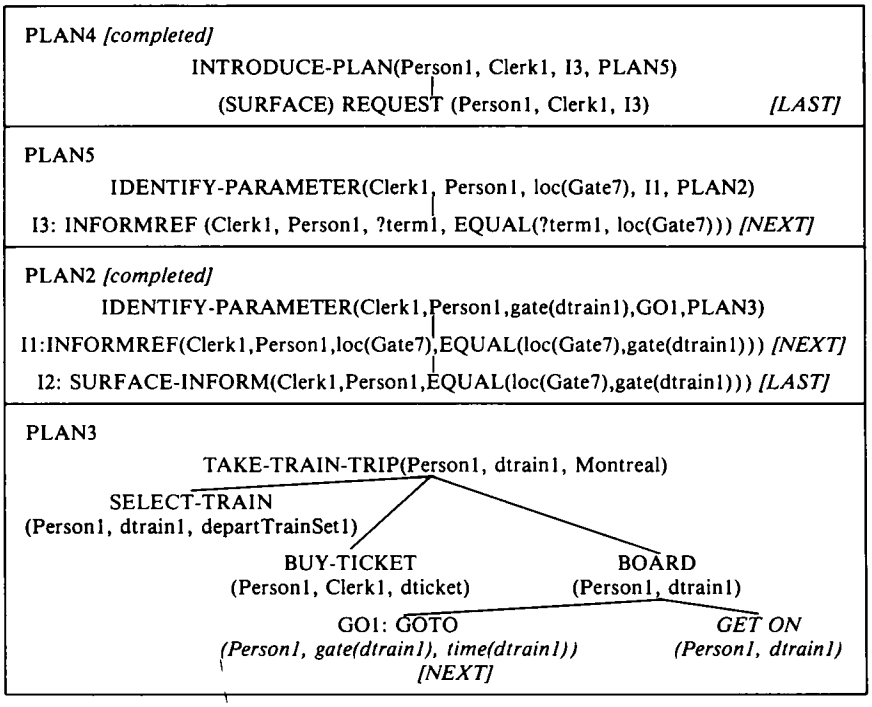


Figure 13. The plan stack after the passenger's second utterance.

the original domain plan involves no communication, that is, there are no further utterances that can execute the domain plan.

### 3.3. Dialogue 3

This section illustrates the system's processing of Dialogue 3 (repeated below for convenience):

User: Show me the generic concept called "employee."
System: OK. <System displays network>
User: No, can you move the concept up?
System: Yes. <System redisplays network>
User: OK, now make an individual employee concept whose first name is "Sam and whose last name is "Jones."

The example illustrates the recognition of a new discourse plan, the treatment of indirect speech acts, the domain independence of the approach (although new domain plans must be introduced, the recognition procedure and the discourse plans remain unchanged), and the applicability of the model to task-oriented as well as information-seeking dialogues (for example, Dialogues 3 and 2, respectively).

In this dialogue the user interacts with a KL-ONE database system that is capable of graphically displaying KL-ONE concepts organized in a taxonomic network (KL-ONE (Brachman et al., 1979) is a knowledge representation language). Figure 14 presents the relevant domain plan schemas for this example. They are taken from Sidner and Israel (1981), with minor modifications and consist of plans to add new data into the network and to examine parts of the network. Both of these have a subplan involving the plan CONSIDER-ASPECT, in which the user considers some aspect of a network, for example by looking at it (the decomposition shown), listening to a description, or thinking about it. Again, the representation of action and time is greatly simplified.

| | |
|---|---|
| HEADER: | ADD-DATA(user, netpiece, data, screenLocation) |
| DECOMPOSITION: | CONSIDER-ASPECT(user, netpiece)<br>PUT(system, data, screenLocation) |
| HEADER: | EXAMINE(user, netpiece) |
| DECOMPOSITION: | CONSIDER-ASPECT(user, netpiece) |
| HEADER: | CONSIDER-ASPECT(user, netpiece) |
| DECOMPOSITION: | DISPLAY(system, user, netpiece |

Figure 14. Graphic editor domain plans.

The processing begins with the parser analysis of "Show me the generic concept called 'employee,'"

SURFACE-REQUEST (user, system, DISPLAY (system, user, E1))

where E1 stands for "the generic concept called 'employee.'" Since there is no stack, the plan recognizer tries to introduce a plan according to preference (3). Through forward chaining, the system finds that the utterance is a REQUEST that introduces some plan. From constraint satisfaction we know that the plan introduced must contain the display action, and recognition from the DISPLAY now occurs. Since the display action could be a step of the CONSIDER-ASPECT plan, which itself could be a step of either the ADD-DATA or EXAMINE plans, the domain plan remains ambiguous. Note that heuristics can not eliminate either possibility, since at the beginning of the dialogue any domain plan is a reasonable expectation. Chaining halts at this branch point and all plans are expanded (again shown by dotted lines). Their effects are asserted, and two stacks constructed as shown in Figure 15.

Again, the planning the system performs is actually simulated based on the dialogue. Since the REQUEST (and thus plan introduction) is completed, in each stack the stack can be popped and the execution of DISPLAY

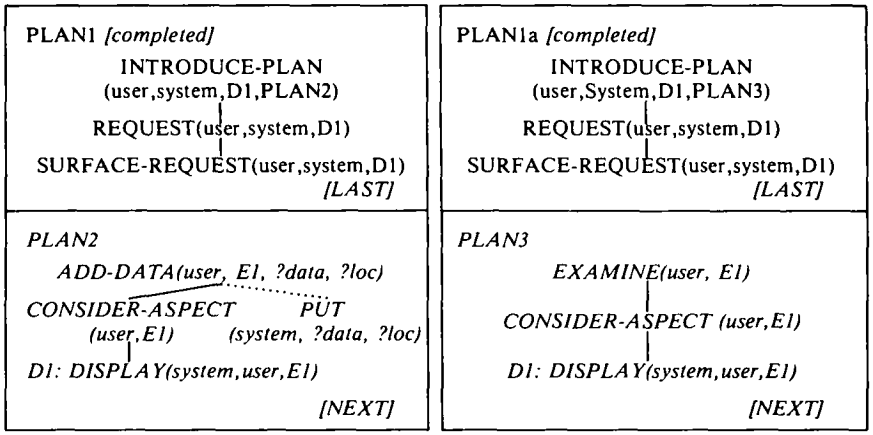


Figure 15. The two plan stacks after the first utterance.

performed as a coherent next move. The system chooses to perform the display even though the reason for this action is still ambiguous. Deciding exactly what to do in such cases is an interesting planning issue. The system also chooses to generate "OK" to explicitly mark the step's completion.

Unfortunately, the system chooses a display that does not allow room for the insertion of a new concept, leading to the user's response "No, can you move the concept up?" In other words, the fact that the system decided to perform the DISPLAY without knowing whether a PUT would follow (and if so what would be PUT and where) has now caused problems in the user's original plan, since the location of the node for the generic concept did not leave enough room to put a new node below it. The utterance is parsed and input to the plan recognizer as the clue word "no" (using the plan recognizer's list of standard linguistic clues) followed by

SURFACE-REQUEST(user, system, I1:INFORMIF (system, user
CANDO (system, M1:MOVE(system,E1, up)))).

(again assuming the resolution of "the concept" to E1). The utterance is recognized by the plan recognizer as either an indirect REQUEST for the system to perform the move or as a literal REQUEST for the system to tell the user if the system can perform the move. Since neither is yet connected to any domain or discourse plan, both alternatives are pursued in both contexts postulated above. (The incremental plan recognition process holds for plan branch points, but not for speech act recognition). Using the knowledge that "no" typically does not signal a topic continuation, the plan recognizer first modifies its standard mode of processing, that is, the assumption that the REQUEST is a continuation (preference 1) is overruled. Note, however, that even without such a linguistic clue recognition of a plan continuation

would have ultimately failed, since neither informing nor moving (that is, the actions requested in the direct and indirect interpretations, respectively) are steps in PLAN2 or PLAN3. The clue thus allows the system to reach reasonable hypotheses more efficiently, since unlikely inferences are avoided.

Proceeding with preference (2) and the indirect speech act hypothesis, the system postulated that either PLAN2 or PLAN3 is being corrected. Since the REQUEST matches both decompositions of CORRECT-PLAN, there are two possibilities:

CORRECT-PLAN(user, system, ?laststep, M1, ?nextstep, ?plan), and
CORRECT-PLAN(user, system, ?laststep, ?newstep, M1, ?plan),

where the variables in each will be bound as a result of constraint and prerequisite satisfaction from application of the heuristics. For example, candidate plans are only reasonable if their prerequisites were true, that is (in both stacks and corrections) WANT(system, ?plan) and LAST(?laststep, ?plan). Assuming the plan was executed in the context of PLAN2 or PLAN3 (after PLAN1 or PLAN1a was popped and the DISPLAY performed), ?plan could only have been bound to PLAN2 or PLAN3, and ?laststep bound to D1. Satisfaction of the constraints eliminates the PLAN3 binding, since the constraints indicate at least two steps in the plan, while PLAN3 contains a single step described at different levels of abstraction. Satisfaction of the constraints also eliminates the second CORRECT-PLAN interpretation, since STEP(M1, PLAN2) is not true. Thus only the first correction on the first stack remains plausible, and in fact, using PLAN2 and the first correction the rest of the constraints can be satisfied. In particular, the bindings yield

(1) STEP(D1, PLAN2)
(2) STEP(P1, PLAN2)
(3) AFTER(D1, P1)
(4) AGENT(M1, system)
(5) ~CANDO(user, P1, PLAN2)
(6) MODIFIES(M1, D1)
(7) ENABLES(M1, P1)
where P1 stands for PUT(system, ?date, ?loc).

Finally, the effects are not already true for our now instantiated hypothesis CORRECT-PLAN(user, system, D1, M1, P1, PLAN2), resulting in its verification. In contrast, no CORRECT-PLAN interpretation can be verified in the case of the literal speech act interpretation (i.e., the REQUEST to INFORMIF). This is because the INFORMIF action satisfies neither the constraint STEP (I1, PLAN2) in the second CORRECT-PLAN interpretation, nor the constraints MODIFIES or ENABLES in the first. Other discourse

plan interpretations connecting the literal speech act with PLAN2 or PLAN3 are not pursued, since they are less preferred due to the clue "no" together with the coherence heuristics.

The effects of CORRECT-PLAN are asserted (M1 is inserted into PLAN2 and marked as NEXT) and CORRECT-PLAN is pushed on the stack suspending the plan corrected, as shown in Figure 16. Note that unlike the first utterance, the domain plan referred to by the second utterance is found in the stack rather than constructed. Using the updated stack, the system can then pop the completed correction and resume PLAN2 with the new (next) step M1.

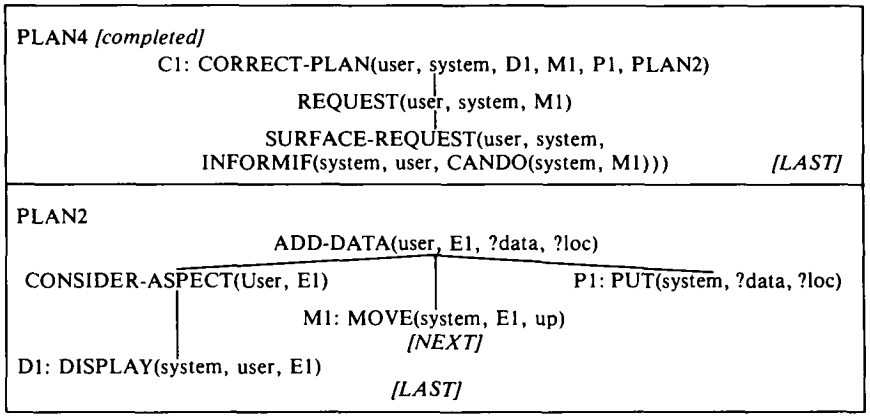


Figure 16. The plan stack after the user's second utterance.

The rest of the dialogue continues in the obvious way. The system can execute act M1, moving E1 up and then the dialogue continues the plan in the expected manner, with the user requesting that the system create a new concept (a specialization of the step P1).

As discussed fully in Litman (1985), our plan recognition framework also proves useful for the analysis of several variations of this dialogue. For example, Figure 17 illustrates the analysis when the user's "No, can you move the concept up?" is preceded by the exchange

User: I can't fit a new individual concept below it.
System: What do you want me to do?

In this case the user's correction is achieved via CORRECT-PLAN's second decomposition, and the system prompts for an elaboration. Now if the user utters "Can you move the concept up?" in this context, the utterance would be a continuation of PLAN6, illustrating how a given utterance can be analyzed differently depending on the discourse context.

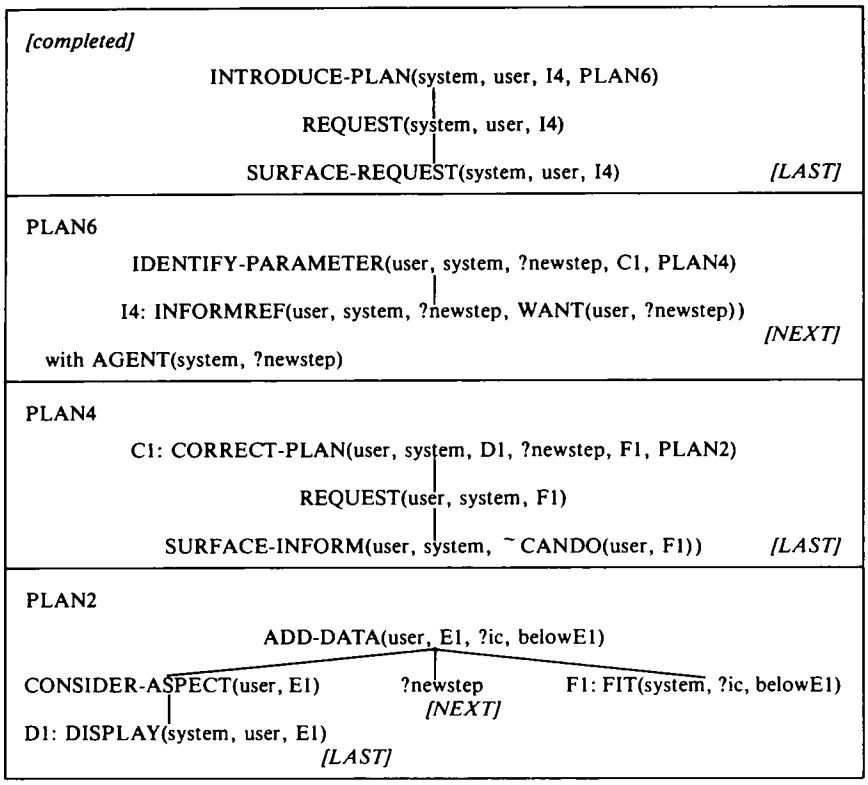


Figure 17. Analysis of a dialogue variation.

## 4. COMPARISONS TO RELATED WORK

### 4.1. Plan Recognition

The major difference between our present approach and previous plan recognition approaches to speech act understanding (e.g., Allen & Perrault, 1980) is that our model has a hierarchy of plans, whereas all the actions in Allen and Perrault are contained in a single plan. (We are excluding differences arising from our concern with dialogues as opposed to Allen and Perrault's concern with single utterances). Our treatment enables the simplification of the notion of what a plan is and the solving of a puzzle that arises in the one-plan systems. Plans recognized in the one-plan systems are action and state descriptions linked by prerequisite, effect, and decomposition relationships, plus a set of knowledge-based relationships. This latter set is not categorized as either a causal or a subpart relationship and so needs a special mechanism. Incorporating this into the action definitions would require having a knowledge precondition for every term in every action. The problem is that these

relationships are not part of any plan itself, but a relationship between plans. In our system this relationship between plans is explicit, eliminating the need for the plan recognizer's special mechanism. For example, the "knowref," "know-pos" and "know-neg" relations of Allen and Perrault are modeled as constraints between a plan and a discourse plan, that is, the plan to perform the task and the plan to obtain (i.e., clarify) the knowledge necessary to perform the task. Our work thus argues for the existence of not only domain intentions, but also discourse intentions indicating how each utterance relates to the domain intentions.

Wilensky (1983) shows how a different type of meta-planning knowledge can be used by plan-based natural language systems. For Wilensky, meta-planning knowledge is defined to mean general knowledge regarding a planner's goals, used during the construction process of any plan. For example, an instance of meta-planning knowledge is the fact that one way a planner can resolve a goal conflict is to abandon the less important goal. Such knowledge is necessary to explain observations that arise from the interaction of multiple goals. Although the taxonomies of Wilensky's work are excellent, the particular details of his theory are never well-specified. For example, Wilensky does not address details of coordination between meta and non-meta plan recognition. This is in contrast to the current work, where the theory specifies the recognition of a discourse plan from every utterance, and the recognition of any other plans via constraint propagation. Wilensky's work also differs from our work in matter of emphasis. Wilensky's meta-plans are used to handle issues of concurrent goal interactions, while the discourse (meta) plans of our work are used to handle issues of discourse.

Finally, although less directly related to our work, it should be noted that several researchers (Cohen & Levesque, 1985; Kautz, 1985; Kautz & Allen, 1986) are currently working on formal foundations for plan recognition. Cohen and Levesque use a theory of rational interaction, expressed in a logic based loosely on possible-worlds semantics, to derive the basis of a theory of communication. Kautz is concerned with formalizing the process of plan recognition by concentrating on the nonmonotonic aspects of such reasoning. Pollack (1986a,b) is also concerned with improving the foundations of plan recognition systems. She reformulates plans as mental phenomena, allowing more general forms of plan inference such as the inference of invalid plans. Our work has implicitly assumed appropriate inputs and has thus not yet had to address this issue.

### 4.2. Plan Recognition and Discourse

Grosz (1977) notes that since in task-oriented dialogues discourse structure follows the task structure, task plan structure can be used (along with surface linguistic phenomena) to construct an underlying discourse structure.

Grosz, however, is primarily concerned with how the recognized discourse structure can then be used to account for linguistic phenomena. Explication of the recognition process is not of primary concern.

Sidner (1985) and Sidner and Israel (1981) outline an approach that extends Allen and Perrault in many of the directions pursued in our work. For example, Sidner is concerned with understanding dialogues both through the use of plans and through the use of surface linguistic phenomena. With respect to the process of plan recognition, Sidner suggests a solution to a class of subdialogues that correspond to debugging the plan in the task domains, by allowing utterances to talk about, rather than always be a step in, task plans. Her work allows for multiple plans to be recognized from a given utterance, but does not appear to relate such plans in a systematic way. For example, in Sidner's system an utterance such as "I can't x" triggers a blocked plan for $x$, as well as a debugging of $x$ that temporarily suspends the blocked plan. However, further details of how these two plans are related and how the suspension is managed are left unexplored. Our model grows out of many of Sidner's suggestions as well as earlier work on the ARGOT system (Allen et al., 1982). Our work details the recognition of multiple plans and their relationships, and provides a mechanism for managing the debugging subdialogues Sidner discusses as well as other forms of clarification, correction, and topic change. We use constraints to explicitly specify discourse/domain plan relationships, and constraint satisfaction to initiate recognition of a domain plan from a discourse plan. Finally, with respect to linguistic clues, Sidner's plan recognition system is never actually coordinated with the recognition of clues. Sidner does, however, claim that interrupting relationships between multiple plans cannot be recognized without such clues (with one exception based on turn-taking). In contrast, our work specifies a model of clue/plan recognition interaction, and shows how to proceed in a purely plan-based manner when clues are unavailable.

Carberry's TRACK system (1983) is an incremental plan recognition system that can hypothesize and track task-level goals during information-seeking dialogues. It is part of a pragmatic component for a robust natural language interface (Carberry, 1986). TRACK extends the work of Allen and Perrault (1980) by both processing dialogues rather than single utterances and manipulating more complex domain plans. Carberry develops a set of heuristics based on principles for coherently shifting focus, since in information-seeking dialogues the underlying plan is not executed during the dialogue and focus shifts are thus not tightly constrained by the task structure. This emphasis on moving around in rather than following a task structure involves ideas similar to our classes of coherency. Unlike our work, however, Carberry's tracking mechanism is solely plan-based, that is, it does not use any linguistic clues to aid its recognition task. Furthermore, information is gathered with respect to a single plan, so there is no concern for recognition and suspension of multiple plans.

Grosz (1979), Levy (1979), and Appelt (1985) extend the planning framework to incorporate multiple perspectives, including both discourse and task goal analysis. However, they do not present details for dialogues. ARGOT (Allen et al., 1982) attempted to extend such work to dialogue processing, and led to the development of much of what has been presented here.

Grosz and Sidner (1986) argue that to adequately explain several discourse phenomena, discourse structure must explicitly be broken down into at least three distinct (but interacting) components: the structure of the sequence of utterances, the structure of intentions conveyed, and the attentional state at any given point. The theory presented here seems to be consistent with many of Grosz and Sidner's ideas. For example, the three components of Grosz and Sidner's theory appear to have loose correlates in the current work's dialogues, set of recognized plans (although Grosz and Sidner do not distinguish between domain intentions and discourse intentions), and the stack of such plans. The matter of emphasis is quite different, however. Grosz and Sidner do not yet address the process of actually recognizing their structures of intentions (e.g., identifying the plan algorithm and the information it uses), or show how the theory could be used to construct a discourse processing system.

### 4.3. Linguistic-Based Approaches to Discourse

The identification and specification of sets of linguistic relationships between utterances forms the basis for many computational models of discourse (Cohen, 1983; Hobbs, 1985; Mann, 1984; McKeown, 1985; Reichman-Adar, 1984). By limiting the relationships allowed in a system and the ways in which relationships coherently interact, efficient mechanisms for understanding and generating well-organized discourse have been developed. Unfortunately, while the relationships proposed in such models seem intuitive (e.g., "elaboration"), there has been little agreement on what the set of possible relationships should be, or even if such a set can be defined. Furthermore, since the formalization of the relationships has proven to be an extremely difficult task, the theories typically depend on unrealistic computational process. As discussed fully in Litman (1986a), our model can be seen as a plan-based alternative to such theories, providing a computationally feasible framework for both representing as well as recognizing implicit relationships between utterances.

## 5. SUMMARY

We have presented a model for understanding subdialogues in conversation, based on the recognition of a hierarchy of domain and discourse plans. The model accounts for subdialogues such as clarification, corrections, and topic change, yet maintains the computational advantages of the plan-based approach.

Knowledge about discourse was incorporated into a planning framework using the same plan-based formalism previously used for representing domain knowledge. In particular, discourse knowledge was formalized using a set of discourse plans, plans explicitly encoding relationships between utterances and domain plans.

Domain and discourse plans were recognized by a context dependent, heuristic plan recognition algorithm. The algorithm was applied after every utterance, and updated the stack of plans representing the current discourse context. First, a discourse plan was recognized from every utterance. In the absence of linguistic clues, the recognition of this discourse plan was constrained by preferring a continuation of a plan on the stack over a clarification, and a clarification of a plan on the stack over a topic switch. The formal constraints specifying the discourse and domain plan relationships were then used to generate recognition of the domain plan from recognition of the discourse plan. In particular, the plan recognizer either matches the discourse plan to a domain plan already on the stack, or constructed a new domain plan for the discourse plan to be about. Finally, the new set of plans were pushed onto the stack. Each discourse plan referred to the plan directly below it as its topic, with the domain-dependent task plan at the bottom.

Our theory was demonstrated by detailing the processing of two dialogues in different domains. The examples illustrated the plan-based processing of clarification, correction, and topic change subdialogues.

## REFERENCES

Allen, J.F., Frisch, A.M., & Litman, D.J. (1982). ARGOT: The Rochester dialogue system. *Proceedings of the National Conference on Artificial Intelligence,* Pittsburgh, PA.

Allen, J.F., Giuliano, M., & Frisch, A.M. (1984). *The HORNE reasoning system* (Tech. Rep. No. 126 revised). Rochester, NY: University of Rochester.

Allen, J.F., & Perrault, C.R. (1980). Analyzing intention in utterances. *Artificial Intelligence, 15,* 143–178.

Appelt, D.E. (1985). *Planning English sentences.* New York: Cambridge University Press.

Brachman, R.J., Bobrow, R.J., Cohen, P.R., Klovstad, J.W., Webber, B.V., & Woods, W.A. (1979). *Research in natural language understanding: Annual report* (Tech. Rep. No. 4274). Cambridge, MA: Bolt, Beranek, and Newman.

Brown, J., & Burton, R. (1977). *Semantic grammar: A technique for constructing natural language interfaces to instructional systems* (Tech. Rep. No. 3587). Cambridge, MA: Bolt, Beranek, and Newman.

Carberry, M.S. (1983). Tracking user goals in an information-seeking environment. *Proceedings of the National Conference on Artificial Intelligence,* Washington, D.C.

Carberry, M.S. (1986). *Pragmatic modeling in information system interfaces* (Tech. Rep. No. 86-07 and PhD Thesis). Newark, DE: University of Delaware.

Cohen, R. (1983). *A computational model for the analysis of arguments* (Tech. Rep. No. 151 and PhD Thesis). Toronto, Ontario: University of Toronto.

Cohen, P.R., & Levesque, H.J. (1985). Speech acts and rationality. *Proceedings of the Meeting of the Association for Computational Linguistics,* Chicago, IL.

Cohen, P.R., & Perrault, C.R. (1979). Elements of a plan-based theory of speech acts. *Cognitive Science, 3,* 177–212.

Cohen, P.R., Perrault, C.R., & Allen, J.F. (1982). Beyond question answering. In W. Lehnert & M. Ringle (Eds.), *Strategies for natural language processing.* Hillsdale, NJ: Erlbaum.

Fikes, R.E., & Nilsson, N.J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence, 2,* 189–208.

Goodman, B.A. (1984). *Communication and miscommunication* (Tech. Rep. No. 5681). Cambridge, MA: Bolt, Beranek, and Newman.

Grosz, B.J. (1977). *The representation and use of focus in dialogue understanding* (Tech. Rep. No. 151). Menlo Park, CA: SRI.

Grosz, B.J. (1979). Utterance and objective: Issues in natural language communication. *Proceedings of the International Joint Conference on Artificial Intelligence,* Tokyo.

Grosz, B.J., & Sidner, C.L. (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics, 12,* 175–204.

Hobbs, J.R. (1985). *On the coherence and structure of discourse* (Tech. Rep. No. CSLI-85-37). Stanford, CA: Stanford University.

Horrigan, M.K. (1977). *Modelling simple dialogs* (Tech. Rep. No. 108). Toronto, Ontario: University of Toronto.

Kautz, H.A. (1985). *Towards a theory of plan recognition* (Tech. Rep. No. 170). Rochester, NY: University of Rochester.

Kautz, H.A., & Allen, J.F. (1986). Generalized plan recognition. *Proceedings of the National Conference on Artificial Intelligence,* Philadelphia, PA.

Levy, D. (1979). Communicative goals and strategies: Between discourse and syntax. In T. Givon (Ed.), *Syntax and semantics (Vol. 12).* New York: Academic.

Litman, D.J. (1985). *Plan recognition and discourse analysis: An integrated approach for understanding dialogues* (Tech. Rep. No. 170 and PhD Thesis). Rochester, NY: University of Rochester.

Litman, D.J. (1986a). Linguistic coherence: A plan-based alternative. *Proceedings of the Meeting of the Association for Computational Linguistics,* New York, NY.

Litman, D.J. (1986b). Understanding plan ellipsis. *Proceedings of the National Conference on Artificial Intelligence,* Philadelphia, PA.

Mann, W.C. (1984). Discourse structures for text generation. *Proceedings of Coling84,* Stanford, CA.

McKeown, K.R. (1985). *Text generation: Using discourse strategies and focus constraints to generate natural language text.* New York: Cambridge University Press.

Polanyi, L., & Scha, R.J.H. (1983). The syntax of discourse. *Text, 3,* 261–270.

Pollack, M.E. (1986a). *Inferring domain plans in question answering* (Tech. Rep. No. MS-CIS-86-40 and PhD Thesis). Philadelphia, PA: University of Pennsylvania.

Pollack, M.E. (1986b). A model of plan inference that distinguishes between the beliefs of actors and observers. *Proceedings of the Meeting of the Association for Computational Linguistics,* New York, NY.

Reichman-Adar, R. (1984). Extended person-machine interfaces. *Artificial Intelligence, 22,* 157–218.

Sacerdoti, E.D. (1977). *A Structure for plans and behavior.* New York: Elsevier.

Searle, J.R. (1969). *Speech acts, an essay in the philosophy of language.* New York: Cambridge University Press.

Searle, J.R. (1975). Indirect speech acts. In P. Cole & Morgan (Eds.), *Speech acts* (Vol. 3). New York: Academic.

Sidner, C.L. (1983). Focusing in the comprehension of definite anaphora. In M. Brady (Ed.), *Computational models of discourse.* Cambridge, MA: MIT Press.

Sidner, C.L. (1985). Plan parsing for intended response recognition in discourse. *Computational Intelligence, 1,* 1–10.

Sidner, C.L., & Bates, M. (1983). *Requirements of natural language understanding in a system with graphic displays* (Tech. Rep. No. 5242). Cambridge, MA: Bolt, Beranek, and Newman.

Sidner, C.L., & Israel, D.J. (1981). Recognizing intended meaning and speakers' plans. *Proceedings of the International Joint Conference on Artificial Intelligence,* Vancouver.

Wilensky, R. (1983). *Planning and understanding*. Reading, MA: Addison-Wesley.