自然言語処理 73−18
(1989. 6. 30)

An Assumption-Based Plan Inference System for Conversation Understanding

John K. Myers

ATR Interpreting Telephony Research Laboratories
Sanpeidani, Inuidani
Seika-cho, Soraku-gun, Kyoto 619-02, Japan
myers%atr-la.atr.junet@uunet.uu.net

Abstract

Dialog understanding is important for machine translation. In order to disambiguate possible translations, it is necessary to represent the perceived beliefs and goals of the dialog participants. However, beliefs cannot be directly observed; thus, when understanding a dialog, it is necessary for a system to make assumptions. The dialog participants must also make assumptions about domain operations and communication, that must be modeled by the system. However, when moving beyond the understanding of simple dialogs to using input from a real corpus, these assumptions can be mistaken. The understanding system therefore must be able to 1) *explicitly model assumptions*; 2) *retract mistaken assumptions*; 3) *automatically retract all beliefs that depend on mistaken assumptions*; 4) *represent multiple possibilities*; 5) *explicitly represent the difference between possible and actual belief.*

This paper presents a plan inference system built using an assumption-based truth maintenance system (ATMS) that accomplishes these requirements. The system can be used as a tool to represent and understand plans based on assumptions and facts. An *assumption* is a possible belief that is treated as believed true, but may be (nonmonotonically) retracted later. A multi-valued *uncertainty logic*, containing the values **actual, possible, hypothetical,** and **inconsistent**, is used to represent assumptions and the degree of belief in assertions' *current* and *predicted* occurrence.

The system uses precondition/effect/decomposition plan schemata with variables in order to represent actions and plans. As examples of how assumptions can be applied to dialog understanding, the assumptions used in understanding illocutionary force behind a speech act are discussed, along with a standard "Take-Trip" problem.

Keywords: Plan Inference, Assumptions, ATMS, Uncertain Reasoning, Belief, Knowledge Representation

1 Introduction

The aim of this work is to embody a plan inference system that explicitly represents and works with assumptions. Such a system provides a tool that can be used for dialog understanding, and for representing and working with problems with uncertain information. Eventually, sometime in the future, a plan inference system will be integrated with a parser and will contribute information to a transfer module and a generator, to form a machine translation system.

Future plan inference systems will work with complex libraries of plans and will have many different types of knowledge. To reach this goal, a necessary first step is this system's *representation* of problems containing assumptions and uncertainties. It is necessary to represent such problems when dealing with non-trivial dialogs where certainty information is limited. In particular, assumptions are useful in dialogs where misunderstandings occur, unresolved multiple possibilities exist for extended periods, multiple correct answers are possible, the system or the people have uncertain beliefs, beliefs can be retracted, diagnosis of possibilities is required, constraints exist between possibilities, or where conversation or understanding is an assumption-based process.

Besides this, the ability to explicitly represent assumptions encourages the examination and incorporation of previously implicit assumptions used in dialog understanding. Without explicit representation, when an assumption is later proved false, there is no way to recognize the problem, recover, and retract the assumption's implications. Currently, most dialog understanding systems start with the assumptions that the hearer and speaker always understand each other perfectly, that they automatically want to cooperate as much as possible, and that they have absolutely no other commitments outside of the conversation. Clearly some of these assumptions can occasionally be incorrect.

The Domain of the Problem. This plan recognition system is designed to understand an ongoing conversation between two people. The system processes the utterances one by one in order, attempting to maintain a representation

of the currently believed concepts as the conversation progresses. The system does not take part in the dialog. The conversation is a task-oriented dialog on the subject of registering for a conference. No "closed-world" assumption is made that the system knows all possible actions.

The Plan Inference Task. Different authors offer different opinions on what "plan recognition" or "plan inference" consists of. This work computes a structure of hierarchical actions with preconditions and effects, along with distinguished goal states. Assumptions are made about the intended meaning of the speaker and the understanding of the hearer, which are also represented as states. Representing this structure serves as recognizing the plan. The representation carefully distinguishes between actual, possible, predicted, and hypothetical occurrences; multiple self-consistent possibilities can be represented. Once the plan has been recognized, high-level information can be sent to a translation or generation program; if such a program requires information to disambiguate a fact, it can query the plan inference system and get justifying explanations as to how much or why a specified concept is believed. In particular, the system can explain the predicted plan that leads to a goal, or give the structure that represents a conversation.

2 Assumptions and Understanding

What Is An Assumption? An assumption is an assertion or concept that is believed by the system. However, rather than being a fact that is believed with certainty, an assumption has the uncertain belief value of **possible**. The system explicitly considers both the case that the assumption is believed true, and the case that the assumption is not believed. In addition, assumptions can be *nonmonotonic*. After an assumption is made, it can later be found to have been mistaken; the assumption can then be *retracted*. The system builds *inference chains* of other concepts that are based on assumptions; the assumptions directly or indirectly imply belief in the inferred concepts. If an assumption is not believed, all the concepts that depend on the assumption are not believed either. Thus, retracting a single assumption can change the belief value of many concepts.

It is possible to have two or more assumptions that are mutually inconsistent. In this case, the system automatically constructs different *possible worlds* for each case [dK86]. After this, whenever a new concept is added to the system, it is automatically added to all relevant possible worlds at the same time.

Why Are Assumptions Necessary for Understanding? Communication is inherently an assumption-based process. People use language as a signal to communicate their ideas. However, it is never completely possible to directly know the concepts of another person. Instead, when attempting to understand a conversation, it is necessary to take a stance and rely on assumptions about the other person's thoughts [Den87]. In a dialog understanding system, there are at least two kinds of assumptions: assumptions that the speaker and hearer make (about the conversation and about domain facts) that must be modeled by the system, and assumptions that the system makes about the speaker and hearer.

In most cases, when the conversation is going well, these assumptions will be valid. However, in cases where the conversation fails temporarily, an assumption will be invalid and must be retracted. One of the people may have a mistaken assumption; the system must model this change in belief. The system may make a mistaken assumption about the dialog or about the participants; this assumption must be changed later. In addition, understanding recovery actions taken by the dialog participants after a mistake is aided by recognizing which assumptions are incorrect. It is necessary to explicitly represent the assumptions used in understanding in order to be able to represent and work with such problems.

3 Presentation of Belief Values

A new logic is presented here, that represents uncertain belief. Instead of **true** and **false**, a concept takes on one of the belief values **actual**, **possible**, **hypothetical**, or **inconsistent**, corresponding to degrees of belief in the concept. In **actual** belief, the belief is certain; the concept is a fact that the system always believes in all possible worlds. **Possible** belief means that the concept is based on an assumption; the system is uncertain as to the degree of belief. The concept is true in at least one possible world, but the system would not be surprised if it turned out to be not true. **Hypothetical** belief means that the system knows about the concept, but has no opinion yet whether it could be believed or not. An **inconsistent** belief means that the concept is certainly not believed (always in all possible worlds).[1]

These belief values are applied to the modals *currently occurring* (the default), and *predicted to be occurring*. Thus, the system can represent **possibly** currently occurring states, states that are **hypothetically** predicted to occur, etc.

The logic is significant in that it represents the degree of uncertainty in the belief of a concept, which is useful information. There is a significant difference between an uncertain possibility that is assumed to be true, even if believed true in all possible worlds, and a certain fact. If there is a problem or a mistake somewhere, it is alright to change an assumption; but it would be surprising to retract a certain fact.

[1] There is logically a fifth, degenerate belief: **Null**, for concepts that the system does not know about.

4 Formal Derivation of Belief Values

The calculations of belief values are built using an ATMS [dK86], which uses an unusual logic. The ATMS operates on terms. A term can be an object constant or a quoted proposition. A term cannot be a functional expression, and cannot contain variables (unless it is a quoted proposition). The variables **a, b, c, d** will be used to refer to arbitrary terms. The ATMS also operates on a finite set of possible worlds. The variable **w** will refer to an arbitrary particular possible world, and $W = \{w1, w2...\}$ to the set of all known (consistent) possible worlds. W can change as worlds get added or retracted by the system. The ATMS has a function **VAL(w,a)** that must return as the value of a term **a** one and only one of the distinguished constants **t** (for **believed-true**) or **u** (for **unknown**)[2] [there is no **believed-false**– see the following Comment]. This is indexed to a particular possible world **w**, and has the interpretation that the given term is believed true or unknown in that possible world. It is thus an opaque modal operator. It is not defined over propositions. The ATMS also has a similar function $IVAL(W, a)$ that depends on all the worlds at the same time, and must return as the "in-ness" value of a term one and only one of the distinguished constants **i** (for **in**) or **o** (for **out**). A term is **in** if, in at least one world, it is valued **believed-true**; otherwise, it is **out**:

$$(IVAL(W,a) = \mathbf{i}) \equiv \exists(w \in W)(VAL(w,a) = \mathbf{t}) \qquad (IVAL(W,a) = \mathbf{o}) \equiv \forall(w \in W)(VAL(w,a) = \mathbf{u})$$

The ATMS also has a function $PVAL(a)$ that must return as the "permanent" value of a term one and only one of the distinguished constants **s** (for **specified-true**), **u** (for **unknown**), or **n** (for **nogood**). These affect the VAL in the following way:

$$(PVAL(a) = \mathbf{s}) \rightarrow \forall W \forall(w \in W)(VAL(w,a) = \mathbf{t})$$

$$PVAL(a) = \mathbf{n}) \rightarrow \forall W \forall(w \in W)(VAL(w,a) = \mathbf{u})$$

The ATMS also has an implication relationship defined over terms, $(a, b, ...c \Rightarrow d)$, that takes one or more commuting antecedents, and a single consequent. This is defined as follows:

$$(a, b, ...c \Rightarrow d) \equiv \forall W \forall(w \in W)((VAL(w,a) = \mathbf{t}) \wedge (VAL(w,b) = \mathbf{t})... \wedge (VAL(w,c) = \mathbf{t})) \rightarrow (VAL(w,d) = \mathbf{t})$$

It can be shown that

$$((((PVAL(a) = \mathbf{s}) \wedge (PVAL(b) = \mathbf{s})... \wedge (PVAL(c) = \mathbf{s})) \wedge (a, b, ...c \Rightarrow d)) \rightarrow (PVAL(d) = \mathbf{s})$$

The $\Rightarrow$ relationship is similarly not defined over propositions. The ATMS also has a distinguished term $\otimes$, known as the "nogood node". The nogood node's permanent value is defined as **nogood**:

$$PVAL(\otimes) \equiv \mathbf{n}$$

which means that its VAL is always **unknown** and its $IVAL$ is always **out**. Any possible world in which $\otimes$ can be proved **believed-true** is inconsistent, and must be retracted from W. It can be shown for a single term a that

$$(a \Rightarrow \otimes) \rightarrow (PVAL(a) = \mathbf{n})$$

Comment. Significantly, because of the referential opaqueness of the functions, there is no way to operate on $\neg a$: i.e., $VAL(w, \neg a)$ is not permitted. It is necessary to define an additional term $a' =$ "$\neg a$" outside of the logic, and then assert the relationship $(a, a' \Rightarrow \otimes)$. However, this is a weak negation; although not both **believed-true**, it is possible for both $VAL(w, a)$ and $VAL(w, a')$ to be **unknown** in the same consistent world, thus not subscribing to the law of the excluded middle.

With this as background, we can now define an interpretation of belief in occurrences of actions and states. The fact that a state or action is occurring is a quoted proposition represented by a term, e.g. a= "The guest is greeting the office". The opaque belief operator $BEL(W, a)$ takes on one of the values **a** (for **actual**), **p** (for **possible**), **h** (for **hypothetical**), or **c** (for **inconsistent**, or contradiction). It is defined as follows:

$$(PVAL(a) = \mathbf{s}) \rightarrow \forall W(BEL(W,a) = \mathbf{a})$$

$$(PVAL(a) = \mathbf{u}) \wedge (IVAL(W,a) = \mathbf{i}) \rightarrow (BEL(W,a) = \mathbf{p})$$

$$(PVAL(a) = \mathbf{u}) \wedge (IVAL(W,a) = \mathbf{o}) \rightarrow (BEL(W,a) = \mathbf{h})$$

[2] *unknown* is equivalent to **not believed-true**; it is *not* equivalent to **believed false**.

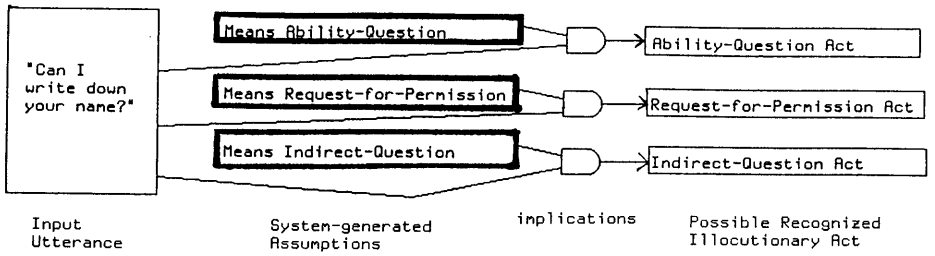

Figure 1: Example: Representing Possible Illocutionary Force by Using Assumptions

$$(PVAL(a) = \mathbf{n}) \rightarrow \forall W(BEL(W, a) = \mathbf{c})$$

In practice, since there is only one current set of all known possible worlds W, the W will be understood and the notation $BEL(a)$ will be used. The combination of these new belief values with the $\Rightarrow$ implication relationship results in the following truth table (a and b commute):

$BEL(a)$	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a
$BEL(b)$	a	a	a	a	p	p	p	p	h	h	h	h	c	c	c	c
$BEL(c)$	a	p	h	c	a	p	h	c	a	p	h	c	a	p	h	c
$(a, b \Rightarrow c)$	T	F	F	F	T	T	F	F	T	T	T	T	T	T	T	T

$BEL(a)$	p	p	p	p	p	p	p	p	p	p	p	p	h	h	h	h	h	h	h	h	c	c	c	c
$BEL(b)$	p	p	p	p	h	h	h	h	c	c	c	c	h	h	h	h	c	c	c	c	c	c	c	c
$BEL(c)$	a	p	h	c	a	p	h	c	a	p	h	c	a	p	h	c	a	p	h	c	a	p	h	c
$(a, b \Rightarrow c)$	T	T	F^*	F^*	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T

Predicted belief values are represented using the same formalism. However, for each term a, a new term a_p = "*predicted*(a)" is created. Then, $BEL(a_p)$ is interpreted to mean "the value of the belief in the prediction of a".

Implementation. The ATMS directly implements the {term, $\Rightarrow$} logic discussed here. The ATMS has two data structures, *ATMS-nodes* and *ATMS-implications*, that directly represent terms and the $\Rightarrow$ relationship, respectively. The VAL, IVAL, and PVAL functions are also implemented directly, as is the $\otimes$ term. The BEL operator is trivially derived from these. Prediction and negation are implemented by appropriately duplicating the ATMS-node. For more information on the ATMS, see [Mye89].

5 Example Application: Representation of Illocutionary Force

Illocutionary force requires assumptions because the speaker could possibly mean any one of several different things when an utterance is stated.[3] Understanding the illocutionary force behind an utterance consists of recognizing that a particular illocutionary act has taken place. This is done in the system by assuming the meaning behind an utterance; the conjuction of the assumed meaning plus the actual utterance act implies that the illocutionary act has occurred.

For example, take the utterance "Can I write down your name?". This could have three possible meanings: it could be a simple question concerning ability (the most literal interpretation); it could be a request for permission; or, it could be an indirect question for the information. Assuming the first meaning is true, the person has just performed an Ability-Question illocutionary act. The conjunction of the second meaning and the utterance act implies the occurrence of a Request-For-Permission act. Finally, if the third possible meaning is believed, in conjunction with the actual utterance it implies belief in the possibility that an Indirect-Question-Act has occurred.[4] (see Figure 1).

Once these assumptions have been explicitly represented, the system can use them as justifications, retract inconsistent assumptions, and work with them in other ways.

6 Plan Recognition

Representation and System Input. Information is represented inside actions, states, and the system input, in Prolog-like forms. The forms consist of constants, variables, and lists of forms. Variables stand for forms. Forms are

*In certain cases where a and b are disjoint (mutually but not absolutely inconsistent), these can be T.

[3] In addition, sometimes utterances can purposefully have more than one meaning. The system can represent both mutually exclusive interpretations and dual interpretations. However, it must know hypothetically which are which ahead of time.

[4] This example has been simplified for illustration purposes. For example, the mutually inconsistent interpretation constraints and the preconditions have been left out.

matched against other forms, and the resulting binding is used to instantiate still other forms (e.g. in plans). All the variables in a form must bind to the same form, except for a special "wild" variable that matches anything. As in Prolog, there are no (explicit) universal or existential quantifiers, and the positions of terms have no inherent meaning apart from that defined by the plans that match and use the forms (i.e., there are no true "operators" or "functions", only relationships).

The specific system representation of forms is not significant, as the system simply stores and matches forms inside the system's states and plan schemata. The plan inference and assumption capabilities of the system do not depend on the type of form used. This particular type of form was chosen only to demonstrate the system; given a matcher, it would be a relatively straightforward matter to convert the system to use another form representation.

The input to the initialized system consists of a list of forms that are asserted sequentially. The forms are intended to represent the surface syntactic meaning of sequential utterances in observed conversations; they are generated by hand, representing the output from a semantic parser. Two categories are used: Ask, for anything ending in a question marker, and Tell, for everything else. The system input forms have no variables.

Representation of States. A state is simply a form (with associated bookkeeping information). It represents the occurrence of the information or the concept represented by the form.

Representation of Actions. Actions are represented by plan schema. An action has a name, an argument list of variables, a list of preconditions, a list of effects, and a list of decompositions. Any of these attributes can be left out, in which case it is assumed null. The following example shows a typical plan schema used by the system.

```
(action     TAKE-TRIP  (?Actor ?Train ?Destination)
  (prec (DESTINATION ?Train ?Destination))
  (eff  (In ?Actor ?Destination))
  (dec  (PURCHASE ?Actor Clerk (Ticket ?Train))
        (GO-TO ?Actor (Location ?Train))
        (GET-ON ?Actor ?Train)  )
)
```

An action can have multiple decomposition lists; in this case, the action can decompose into the members of any one list. Thus, decompositions are conjunctions, and multiple decompositions represent disjunctions. Both the preconditions and the effects are forms representing states, while the decomposition forms can be states or can be other actions. The decompositions are temporally unordered.

The system is initialized by reading in a list of (plan schema) actions. At this point, the actions' variables are unbound. The system binds the variables and compiles or interprets the schemata into an internal representation.

6.1 Recognition, Prediction, and Inference

Plan **recognition** as defined here consists of recognizing the high-level actions, plans and states that are actually or possibly occurring, based on the actual occurrence of low-level actions. It operates in a bottom-up, forward-chaining manner. Plan **prediction** consists of starting with actual and possible goals, and recognizing possible predicted plan steps, using spreading activation in a top-down, backward-chaining manner. Plan **inference** consists of matching predicted actions against possible or actual actions, and forming a chain of possible predicted actions between an actual action and an actual goal.

In order to instantiate a full plan inference system on an ATMS, two steps are involved. The first step is the creation of a model or models for working with the plans, that are expressed in terms of inferences. The second step is instantiating this inference model using the primitives provided by the ATMS.

Inference Models for Plans. The system uses three inference models: one for plan recognition, one for prediction, and one for plan inference.

The plan recognition model takes a strong view of recognition. All of the terms in a decomposition must be (possibly or actually) present before an action can be recognized. In addition, all of the preconditions must be (possibly or actually) present. Thus, the presence of the action is inferred from the conjunction of the decomposition terms and the precondition terms. This is true as well for each alternative decomposition; thus, the action is implied by a disjunction, in which each term is a conjunction representing an alternative decomposition. In practice, since most of the preconditions are mental states, they must be assumed; thus, the decompositions typically contribute the most influence to action recognition. In the inference, if all the terms are facts, i.e. **actually** believed, then the action occurrence is **actually** believed as well. Otherwise, if at least one antecedent (decomposition or precondition) is **possibly** believed (and the rest are **actual**), then the action is only **possibly** believed as well.

In this strong model, no action is taken if at least one of the antecedents only has **hypothetical** belief–the action remains **hypothetical** as well. A weak model might grant the action **possible** belief if at least one decomposition is **possibly** believed (and the rest are **hypothetical**). However, it was felt this would be too unspecific. More research is required on this question.

The recognition model also takes a strong view of effects. If an action is recognized, all of its effects are necessarily asserted. Thus, each effect is inferred from the action. If the action is **actual**, the effect state is **actual**; if the action is **possible**, the effect is **possible**. No allowance is made for alternative effects or effects that fail. However, these can be simulated using assumptions. Recognition thus propagates bottom-up through the decompositions, and in a forward-chaining manner through the effects.

Because *predicted occurrence* is a different concept from *current occurrence*, it requires representation using different assertions: "(PREDICTED *content*)" rather than simply "(*content*)". Each of the prediction assertions can therefore likewise take on the uncertainty values: **actually** predicted, **possibly** predicted, **hypothetically** predicted, or **inconsistent** prediction.

The prediction model, in contrast to the recognition model, uses a weak method similar to spreading activation. Recognition of a goal (actual or possible) sets up an assumption which predicts that that goal will be attempted. Prediction of any one of the effects of an action causes prediction of the action. When an action is predicted, the inference is made that each of the action's preconditions and decompositions is predicted as well. Actual predictions imply actual predictions, and possible predictions imply possible predictions. Prediction thus propagates top-down through the decompositions, and in a backward-chaining manner through the preconditions.

Due to the explanation facilities of the ATMS, each activation spread is labeled with the name of the assumption that originally caused it. Thus, if there are multiple goals, it is possible to query a possibly predicted node as to which assumption causes it to be believed. If more than one goal is contributing to its belief, more than one answer will be returned.

The inference model is very simple: any concept in which both the current occurrence and the predicted occurrence are believed either **possible** or **actual** signals an inferred plan. The plan has been completed up through the current occurrence, and it is predicted to (possibly) continue through the predicted occurrences to the goal that caused the predictions.

Instantiation in the ATMS The inference model shows how the system instantiates plan inference using the ATMS. Each inference is instantiated by an ATMS-implication. Each concept is instantiated by an ATMS-node. The nodes start out as **hypothetical**, and get modified by the user or by value propagation.

7 Operation of the System

The system performs plan inference by instantiating ATMS-nodes and ATMS-implications to represent specific actions and states. The actions or states should have their variables bound before or during instantiation. The nodes can be instantiated in two manners: at initialization (compiled), or as the next utterance is processed (interpreted). In practice, both methods are used.

The system first goes through an initialization phase, where all states and actions that can a priori be supposed to be significant are compiled into the ATMS. Each such state or action is explicitly instantiated by a **hypothetical** ATMS-node. Each implication used to describe an action is instantiated by an ATMS-implication.

Next, the input utterances are asserted one by one. Each utterance introduces new information into the system, which is used to instantiate (if not already there) new hypothetical instances of significant actions. When a specific action is instantiated, its argument list is bound; this binds the rest of the non-wild variables in the action. An instance of a specific action is represented by a single ATMS-node; its decompositions, preconditions and effects are represented by ATMS-implications to other ATMS-nodes, following the inference models, as described in Section 6.1.

However, the instantiated actions and states are still hypothetical. The actual occurrence of a statement then sets the node representing that statement to **actual**, or, if an assumption, to **hypothetical**. The system then propagates the effect of this state change through the existing network, changing belief values as required, in a bottom-up fashion.[5] An assumption retraction operates in a similar manner. Since at this point no matching is done and the system is essentially executing productions on nodes that have already been instantiated, recognition, prediction and plan inference are quite fast. A contradiction imposed on a set of two or more nodes creates mutually inconsistent possible worlds; a contradiction between a possible fact and an actual fact eliminates the possible fact. Alternatives between different interpretations are thus asserted by the system, and resolved by inconsistencies. If a possible goal is recognized by the system, instead of assuming the concept itself, the prediction of the concept is assumed.

Recognition of Plan Inferences It is necessary for the system to recognize when a possible or actual action matches a prediction of the same action. This is most easily done using the explanation facility of the ATMS. The system builds a single special node, "PLAN-INFERRED", with an interrupt routine attached to it. Every time the system creates an action belief node, it also creates a predicted-action belief node for that action, and a third, special "interested in this action" assumption. These three nodes are set to imply the PLAN-INFERRED node. When both the possible action and the predicted action are simultaneously believed, the PLAN-INFERRED node becomes

[5] Actually, in an ATMS each node is not tagged with its belief values, but with a list of possible worlds in which it is believed. The system then automatically propagates these possible worlds. The belief values are derived from this.

believed as well (since the "interested" node is also believed). This triggers the interrupt routine, which uses the explanation facility to find out *why* it is believed, out of the hundreds of implications pointing to it. The routine then reports the plan inference match, and the resulting plan. Finally, since this match has been reported, it is no longer of interest; the routine sets the "interested" node to **nogood**, which disables belief in the PLAN-INFERRED node and re-arms the interrupt.

8 The Train Example

For ease of understanding, a standard example, taken from [All87], is used. Ten actions (Take-Trip, Go-To, Get-On, Give, Find-Seat, Purchase, Inform, Request, Inform-Ref, Inform-If) are used; the system creates 91 ATMS-nodes and 110 ATMS-implications to instantiate the specific concepts used by the system. The input is the following:

```
(Tell Jack Clerk (Want Jack (In Jack Rochester)))
(Request Jack Clerk (Inform-Ref Clerk Jack X (Eq X (Price (Ticket Tr1)))))
```

The output is the following:

```
Plan inferred for (REQUEST JACK CLERK (INFORM-REF CLERK JACK X (EQ X (PRICE (TICKET TR1))))):
POSSIBLE:  (PREDICTED (REQUEST JACK CLERK (INFORM-REF CLERK JACK X (EQ
X (PRICE (TICKET TR1))))))
POSSIBLE:  (PREDICTED (WANT CLERK (INFORM-REF CLERK JACK X (EQ X (PRICE (TICKET TR1))))))
POSSIBLE:  (PREDICTED (INFORM-REF CLERK JACK X (EQ X (PRICE (TICKET TR1)))))
POSSIBLE:  (PREDICTED (KNOW-REF JACK X (EQ X (PRICE (TICKET TR1)))))
POSSIBLE:  (PREDICTED (GIVE JACK CLERK (PRICE (TICKET TR1))))
POSSIBLE:  (PREDICTED (PURCHASE JACK CLERK (TICKET TR1)))
POSSIBLE:  (PREDICTED (TAKE-TRIP JACK TR1 ROCHESTER))
POSSIBLE:  (PREDICTED (IN JACK ROCHESTER))
```

This example is extremely simple. It leaves out many assumptions that are obviously required, e.g. that the Clerk chooses to want to respond to Jack's request (instead of reflexively responding to all requests), and that the Clerk understands what Jack means. Nonetheless, it can be used to illustrate the system's capabilities of: explicitly modeling assumptions; modeling certain vs. uncertain information; retracting mistaken assumptions; automatically retracting dependent beliefs; representing inconsistencies; and representing multiple possibilities.

At this point in the processing, the Clerk **actually** wants to inform Jack of the price. However, suppose it is uncertain whether the Clerk knows the price of the ticket or not (she looks new). In this case, the KNOW-REF concept is *assumed*, and takes on a **possible** belief. Does the Clerk tell Jack the price of the ticket, and does Jack learn the price? Since knowing the price is a necessary precondition, in one possible world, the Clerk tells Jack the price, and he learns it; in another possible world, she doesn't and he doesn't. Thus, the belief in (KNOW-REF JACK X (EQ X (PRICE (TICKET TR1)))) is **possible**, based on propagation from the Note that in a normal plan recognition system, there is no way to explicitly represent and work with assumptions and uncertain information. Either something is happening, or it isn't. Also, there is no way to propagate the effects of uncertainty: if something happens, then its effects certainly happen; if something doesn't happen, then its effects certainly don't happen. Representing that something is possible but not certain provides more information and is a stronger denotational system.

Through propagation, the system now believes it's possible that Jack learns the price of the ticket. However, suppose that the Clerk says, "I'm new; I don't know the ticket price." and the system asserts that it's certain that `(Not (KNOW-REF CLERK X (EQ X (PRICE (TICKET TR1)))))`. Because the previous Know-Ref concept was an assumption, while the current concept is a certainly, the previous concept is retracted and gets the value **inconsistent**. Systems without uncertainty get into trouble when both a fact and its negation are asserted in the data-base. There is no non-arbitrary way to tell which one should be believed and which one should be retracted.

At this point, the effects of the retraction are now propagated. Although before the system believed the concept that Jack learns the ticket price was **possible**, now it is only **hypothetical**. Normal plan inference systems do not keep track of the necessary bookkeeping associated with propagating retractions.

Suppose now that Jack intends to buy some soda. But, he only brought enough money with him to pay for the ticket. He can buy the soda, or he can buy the ticket, but he cannot do both. If Jack asks the Clerk about the price of the soda, it is necessary to represent this inconsistency in order to understand Jack's plans and the conversation. This can be represented by the mutually inconsistent command `(nogood-set '(Has Jack (Price (Ticket Tr1))) '(Has Jack (Price Soda)))`. This creates one possible world in which "Jack buys the soda" is **possible**, and another possible world in which "Jack buys the ticket" is **possible**. However, *even though both of these assertions are in the data base*, they will never be believed together in the same possible world. Naturally, the effects of this propagate appropriately. Normal plan recognition systems have difficulty representing multiple possibilities and do not represent mutual inconsistencies between concepts.

9 Discussion

The system is designed to represent and work with a particular class of plans, i.e. those composed of actions with preconditions, effects, and decompositions. No claim is made that the system provides a parser or a rigorous syntactic sentence representation, as the research did not concentrate on those aspects. The system is intended to be interfaced with an existing typed feature structure unification-based rewriting system [EZ89] in the near future; naturally, at this time the input interface will have to be replaced.

The system is able to represent uncertain and nonmonotonic belief. Obviously, assumptions are not necessary if all information and beliefs are certain, and no beliefs ever get retracted. In limited cases where such simplifying assumptions can be made successfully, there is no need for the capabilities of this system. However, if there is uncertain information, or beliefs that change, it is necessary to be able to explicitly represent and work with this type of belief. This system provides the means for this representation.

The current system does not completely capture the default nature of the assumptions that people make. Human assumptions are probable, not just possible. In addition, people are able to make implicit assumptions that are not examined if nothing fails, but can be examined and retracted if a mistake occurs, resulting in memory savings. Further research is required to understand and model this phenomenon. A second important area of future research is exploring the trade-off between precompiling inferences (space-expensive) and interpreting inferences (time-expensive) into the ATMS representation. Also, currently temporal representation is deficient, as is nested belief and plan expectation.

A practical system should accept multiple possible surface forms and rank their likelihood. The current system has no method of ranking different possibilities, performing evidential reasoning, or determining the degree of probability of a situation. Thus, it cannot yet disambiguate between multiple possible inputs.

10 Conclusion

The first version of a system that performs plan inference based on assumptions has been discussed here. The system can explicitly represent assumptions used in the plan inference process. The system uses an uncertainty logic to represent possibilities and certainties. Assumptions are nonmonotonic and can be retracted; all inferences depending on the assumption are automatically retracted at the same time. This capability is used to explicitly represent assumptions that were previously implicit in understanding possible illocutionary acts. Since these are represented explicitly, the system can reason about cases when these assumptions break down or are proved incorrect.

Although assumptions are certainly a step in the right direction, it still remains to be seen whether the plan schema formalism is powerful enough to represent the actual information required in dialog understanding and machine translation. This is probably the most significant area of future research, and can only be proven by processing many more actual conversations and understanding the limits of the current formalism.

References

[AI89] Hidekazu Arita and Hitoshi Iida. *Tri-Layered Plan Recognition Model for Dialogue Machine Translation (in Japanese).* Technical Report TR-1-0067, ATR Interpreting Telephony Research Laboratories, Kyoto, Japan, 1989.

[All87] James Allen. *Natural Language Understanding.* Benjamin/Cummings Publising Co., Menlo Park, CA., 1987.

[Den87] Daniel C. Dennett. *The Intentional Stance.* The MIT Press, Cambridge, Mass., 1987.

[dK86] Johan de Kleer. An assumption-based tms. *Artificial Intelligence*, 28(2):127–162, March 1986.

[EZ89] Martin C. Emele and Remi Zajac. *RETIF: A Rewriting System for Typed Feature Structures.* Technical Report TR-1-0071, ATR Interpreting Telephony Research Laboratories, Kyoto, Japan, 1989.

[Gol70] Alvin I. Goldman. *A Theory of Human Action.* Prentice-Hall Inc., Englewood Cliffs, NJ., 1970.

[Kno88] Craig A. Knoblock. Data-driven plan recognition. March 1988. CS Dept., Carnegie-Mellon University, Pittsburgh, PA.

[Mye89] John K. Myers. *The ATMS Manual (Version 1.1).* Technical Report TR-1-0074, ATR Interpreting Telephony Research Laboratories, Kyoto, Japan, 1989.

[Suc87] Lucy A. Suchman. *Plans and Situated Actions.* Cambridge University Press, Cambridge, Mass., 1987.