Logic-based Plan Recognition for Intelligent Help Systems

Mathias Bauer, Gabriele Paul

RR-93-43
Research Report

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Research Report

RR-93-43

Logic-based Plan Recognition for Intelligent Help Systems

Mathias Bauer, Gabriele Paul

October 1993

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Postfach 20 80
67608 Kaiserslautern, FRG
Tel.: + 49 (631) 205-3211
Fax: + 49 (631) 205-3210

Stuhlsatzenhausweg 3
66123 Saarbrücken, FRG
Tel.: + 49 (681) 302-5252
Fax: + 49 (681) 302-5341

Deutsches Forschungszentrum für Künstliche Intelligenz

The German Research Center for Artificial Intelligence (Deutsches Forschungszentrum für Künstliche Intelligenz, DFKI) with sites in Kaiserslautern and Saarbrücken is a non-profit organization which was founded in 1988. The shareholder companies are Atlas Elektronik, Daimler-Benz, Fraunhofer Gesellschaft, GMD, IBM, Insiders, Mannesmann-Kienzle, Sema Group, Siemens and Siemens-Nixdorf. Research projects conducted at the DFKI are funded by the German Ministry for Research and Technology, by the shareholder companies, or by other industrial contracts.

The DFKI conducts application-oriented basic research in the field of artificial intelligence and other related subfields of computer science. The overall goal is to construct systems with technical knowledge and common sense which - by using AI methods - implement a problem solution for a selected application area. Currently, there are the following research areas at the DFKI:

- Intelligent Engineering Systems
- Intelligent User Interfaces
- Computer Linguistics
- Programming Systems
- Deduction and Multiagent Systems
- Document Analysis and Office Automation.

The DFKI strives at making its research results available to the scientific community. There exist many contacts to domestic and foreign research institutions, both in academy and industry. The DFKI hosts technology transfer workshops for shareholders and other interested groups in order to inform about the current state of research.

From its beginning, the DFKI has provided an attractive working environment for AI researchers from Germany and from all over the world. The goal is to have a staff of about 100 researchers at the end of the building-up phase.

Dr. Dr. D. Ruland

Director

Logic-based Plan Recognition for Intelligent Help Systems

Mathias Bauer, Gabriele Paul

DFKI-RR-93-43

This paper also appeared in the Proceedings of the Second European Workshop on Planning (EWSP93).

This work has been supported by a grant from The Federal Ministry for Research and Technology (FKZ ITWM-9000 8).

ISSN 0946-008X

Logic-based Plan Recognition for Intelligent Help Systems

Mathias Bauer and Gabriele Paul
German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany
e-mail: {lastname}@dfki.uni-sb.de

Abstract

Intelligent help systems aim at providing optimal help to the users of complex application systems. In this context plan recognition is essential for a cooperative system behaviour in that it allows the prediction of future user actions, the ascertainment of suboptimal action sequences or even serves as a basis for user-adapted tutoring or learning components. In this paper a new approach to incremental plan recognition based on a *modal temporal logic* is described. This logic allows for an abstract representation of plans including control structures such as loops and conditionals which makes it particularly well-suited for the above-mentioned tasks in command-language environments. There are two distinct phases: With a generalized *abductive reasoning* mechanism the set of valid plan hypotheses is determined in each recognition step. A *probabilistic selection*, based on Dempster-Shafer Theory, then serves to determine the "best" hypotheses, in order to be able to provide help whenever required.

Contents

1 Introduction

Plan recognition is the identification of a user's plans or goals from the available evidence, which comprises his actions as well as, e.g., information about his preferences contained in a user model (see, e.g., [Car90b]). This knowledge of a user's pursued goals is a prerequisite for cooperative behaviour in the context of help systems (cf. [GL92]). Plan recognition components in user interfaces allow, for example, the prediction of future actions which forms the basis for offering semantic plan completion to the user. Furthermore, the recognition of suboptimal user behaviour is a prerequisite for proposing optimal plans in the context of an active help system. Recognizing the sources of a plan execution failure, e.g., by determining an unsatisfied plan precondition, allows for flexible error handling which is adapted to the current context of the user. More cooperative help systems can also be obtained by the incorporation of plan monitoring or tutoring based on plan recognition results. Adding a probabilistic reasoning module makes plan recognition even more effective as it allows taking into account quantitative measures of user preferences and enables the choice of *one* most likely hypothesis whenever required. The approach described in the following sections is a combination of logic-based plan recognition and such a probabilistic selection.

The problem of determining an agent's goals or plans can be viewed as a natural application of abductive reasoning: Plans are, in general, *hypotheses* that are plausible at the current state of knowledge and account for or explain the observed actions carried out to date.[1] The idea behind abduction is a kind of inverted Modus Ponens, i.e., from an occurrence of an observation ω and the rule "φ implies ω", an occurrence of φ is inferred as a *plausible* hypothesis or explanation for ω (see [Gou50], [Pei58]). Thus, abduction is a form of "defeasible" inference, i.e., the formulae sanctioned are just plausible and submitted to verification.

The notion of abduction in the context of plan recognition has only been considered lately and the opinion evolved that all plan recognition is inherently a kind of abductive problem independent of the actual plan recognition model (see [May92]): "The traditional approach to plan recognition, found in seminal works of Wilensky (...) and Allen (...) is to chain together a sequence of abductive inferences" (cf. [All79], [Wil78]).

Using the advantages of a formal logic-based theory—like clear semantics—our approach is based on the modal temporal logic LLP (cf. section 3). This logic has proved to be an effective means of plan recognition as well as of planning (see [BBD+93]) so that one uniform formalism suffices for both tasks. The incorporation of control structures such as conditionals and loops in LLP makes it especially suited to applications in command language environments (see also [BDK92]). As many complex software packages fall into this class, there is a broad range of potential applications in the context of intelligent help systems. The examples given below are taken from the domain of electronic mail under UNIX and provide evidence for the existence of practical applications of our formal framework. On the other hand, the control structures and additional features of LLP like temporal abstraction allow to describe user behaviour in a more concise way and on a more abstract level, which implies that a user can also be given help in a way which enables him to understand better the underlying structure of a plan.

Up to now there are no abductive approaches for modal logics like LLP (see also [BN92]). It is argued in section 2 that temporal modal logics seem to require special

[1] For a detailed overview on abduction see [KKT92], or [Pau93].

criteria for explanations, as the adoption of the predicate logic definition of abduction may result in counterintuitive results.

Goodman and Litman (see [GL92]) present some constraints for plan recognition that should be respected to obtain systems which are on the one hand theoretically well-founded but on the other hand also well-suited to the different plan recognition tasks: The ability to *predict future actions* is an essential component of plan recognizers. To increase efficiency they suggest *incremental recognition* after each observation, using the context to prune the search space in each recognition step. This is what is realized by our plan recognition approach. One further point is the *incorporation of probabilistic or heuristic reasoning* to cope with ambiguity and to allow qualified help. It seems reasonable to *constrain the set of all feasible plan hypotheses* in order to prevent over- or undercommitment or loss of information by adopting a more abstract plan rather than a disjunction.

In order to be able to force a decision among the various hypotheses if, for example, the user directly asks for help to complete his plan, there must be a criterion to judge the "quality" of a plan hypothesis which enables the "best" guess of the actually pursued plan to be made. Such a criterion can be obtained by encoding the knowledge of the user's typical behaviour and the possible impact of new observations in a numerical formalism for dealing with uncertainty and incomplete knowledge, where the numerical values represent the probability of a given hypothesis.

Candidates for such a formalism are—among others—probability theory and Dempster-Shafer Theory (DST). Similar to [Car90a], we will adopt an extended version of DST for our purposes. This choice will be motivated in section 5.

The rest of this paper is organized as follows: After a short motivating example in the next section, we will introduce the logical foundations of our work in section 3. Sections 4 and 5 describe the phases of abductive plan recognition and DST-based selection, respectively. We summarize our work and discuss related approaches in section 6.

2 A Motivating Example

In this section we will give a small example for the recognition of plans taken from the domain of electronic mail. Using this example we want to introduce and explain the basic notions of our plan recognition scenario and furthermore motivate our new approach to abductive recognition and probabilistic selection in a modal temporal logic.

A prerequisite for the recognition of plans[2] is knowledge of a user's possible actions and the combination of these actions in complex action sequences, which describe typical user behaviour. In general, this knowledge is stored in a plan hierarchy as introduced by Kautz, e.g., in [Kau87]. Apart from the information on the *decomposition* of plans the hierarchy also contains information on *abstractions* of hierarchy elements.

For example, the two actions ex(save) and ex(write)—where the predicate ex is used to refer to actions— are both used in a mail system to store a message. Thus they can be summarized by the abstract action ex(store_message). ex(store_message) itself could be a part of the complex action sequence read_and_store, e.g., if this sequence can be decomposed into the actions ex(read) and ex(store_message). In the graphic representation in figure 1 preconditions that might exist are omitted. Abstractions are characterized by

[2] unless so-called *novel plans* shall be recognized (cf. [GL92]).

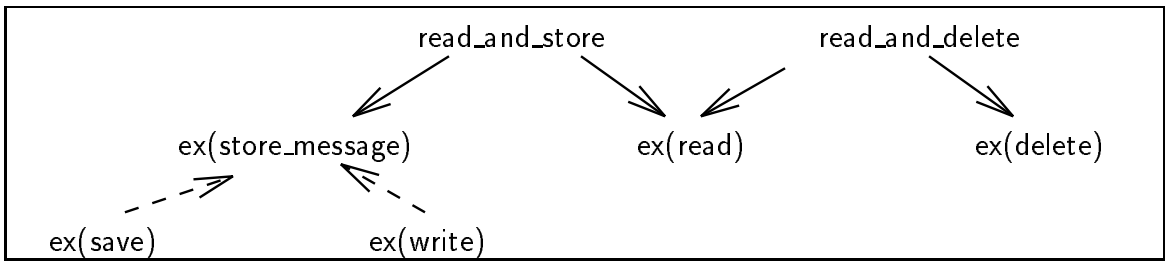

Figure 1: Example of a simple plan hierarchy (PH_1)

dashed arrows, decompositions by solid ones. The representation of the decomposition does not imply an order.

The logical representation of the abstraction hierarchy is

$$\forall x.ex(save(x)) \supset ex(store\_message(x)) \quad (1)$$
$$\forall x.ex(write(x)) \supset ex(store\_message(x)) \quad (2)$$

The decomposition of a plan is given by a plan formula which may also contain preconditions or constraints. Consider the plan read_and_store. A modal formula using the operators $\Diamond$ (read: *sometimes*) and ; (read: *chop*) for temporal abstraction and sequential composition, resp., looks as follows: $\Diamond(ex(read(x)); ex(store\_message(x)))$. Thus the decomposition part of the hierarchy is[3]

$$\forall x[read\_and\_store(x) \supset \Diamond(ex(read(x)); ex(store\_message(x)))] \quad (3)$$
$$\forall x[read\_and\_delete(x) \supset \Diamond(ex(read(x)); ex(delete(x)))] \quad (4)$$

Now suppose the user is seen to execute action ex(read(3)).[4] Intuitively valid explanations for this observation are the assumptions that the user at some time reads and then stores message 3 or that he reads and then deletes it, because ex(read(3)) fits into the corresponding plans $\Diamond(ex(read(x)); ex(store\_message(x)))$ and $\Diamond(ex(read(x)); ex(delete(x)))$ from (3) and (4).

However recalling the "classical" definition of abduction (see also section 4) which requires for sound hypotheses that the observation can be inferred from the hypotheses and the given theory, these plans would not be accepted as explanations (cf. also section 4).[5] In the following, we argue that for this reason, this condition should be weakened, i.e., having observed ex(read(3)), the plans $\Diamond ex(read(3)); ex(store\_message(3))$ and $\Diamond ex(read(3)); ex(delete(3))$ will be assumed to be explanations. Thus we are able to predict the user's next actions and we might give him the option of executing the rest of the plan automatically.

We argue that hypotheses are valid, whenever the original assumptions—in this case $\Diamond ex(read(x)); ex(store\_message(x))$ and $\Diamond ex(read(x)); ex(delete(x))$—can be *refined* by the observation, which means that it is possible to incorporate the observation into the hypothesis by filling the gaps left by temporal or other abstractions such as missing variable instantiations. This refinement corresponds to a selection of those models of the hypothesis in which the observation is true. The formal definition is given in section 4.

[3]Note that the hierarchy in the figure does not reflect the order implied by the *chop*-operator.

[4]Observations will always be sentences of the form $ex(command)$.

[5]Here and in the following we will use the terms "hypothesis" and "explanation" as equivalent.

In general, there may be several possible refinements, as the temporal structure may not be unique. Take for example the plan $\Diamond\, ex(read(x)); ex(store\_message(x)))$ and the observation ex(read(3)). Either the observed read is the one expected in the plan, or we expect the action once again at a later point in time, e.g., with another parameter.

Thus, we have a new condition for valid explanations, but still no method of determining which explanations should be tested for this condition. "Good" hypotheses should satisfy several criteria (cf. [Pei58]): They should not only be verifiable, but also be efficiently computable. In general, the definition of some predefined set of abducibles avoids an explosion of search space and serves to conduct the generation of explanations. For modal logics this is even more important as the language is very expressive. For example, the observation a could give rise to the hypotheses $\Diamond\, a, \Diamond\, b, \ldots, a; b, \ldots$. To overcome this problem we will use quite strong heuristics to guide the construction of refinements in order to reach only those plans which are of interest in plan recognition.

Apart from the modal hypotheses for which we use the refinement condition mentioned above, the plan hierarchy contains also more abstract explanations, for example, the plan read_and_store(3). For the generation of those explanations which contain no modal operators and thus require no refinement, we will make a constructive proof guided by proof strategies following the paradigm of tactical theorem proving as proposed in [Con86], [HRS90], and [BDK92]. The feasibility of this method in our context will also be discussed in section 4.

The abductive mechanism sketched above expresses no preference of one hypothesis over another. If, however, we take into account the user's preferences, we may obtain a criterion for doing so. Suppose we can assign numerical values to the various plan hypotheses which reflect our knowledge of the user's typical behaviour. We will do so by constructing a so-called *basic probability assignment* (bpa) from Dempster-Shafer Theory which will be described in section 5. Such a bpa allows lower and upper bounds for the probabilities of certain hypotheses to be compactly represented without needing to completely specify all possible joint and conditional probabilities. In the above example, we might have the following values:[6] {read_and_store} – 0.1, {read_and_delete} – 0.0, {read_and_store, read_and_delete} – 0.9. This means that we know very little about the user, but that the read_and_store plan is slightly preferred over read_and_delete.

The decomposition part of the plan hierarchy is encoded in a set of weighted rules mapping observations to hypotheses. Assume we have—among others—a rule which states that the occurrence of a read command strongly supports our belief in the user pursuing read_and_store (with strength 0.7) and weakly supports our belief in read_and_delete (with strength 0.3). The application of this rule is allowed because in the above phase of plan recognition both hypotheses were considered feasible explanations for the observation. Its result is a new bpa which assigns the values 0.72 and 0.28 to read_and_store and read_and_delete, respectively (see also section 5). Therefore, if it were necessary to opt for one alternative, the former one would be selected.

3 The Logical Language

The interval-based modal temporal logic LLP (Logical Language for Planning) which is the formal basis of our abductive approach to plan recognition was designed for planning

[6] Here, a non-empty set stands for the disjunction of its elements.

purposes in command language environments (see [BDK92]). LLP is a linear-time logic that is essentially a combination of the temporal logic for programs (cf. [Krö87]) and a first-order version of *choppy logic* from Rosner and Pnueli (see [RP86]). In the following, we will give a short review of the basic concepts.

Let $\Sigma^{\mathcal{F}}$ and $\Sigma^{\mathcal{P}}$ be signatures of function and predicate symbols, respectively. We define $\Sigma = \Sigma^{\mathcal{F}} \cup \Sigma^{\mathcal{P}}$. The set of variables V is determined by $V\mathcal{G} \cup V\mathcal{L}$ with $V\mathcal{G}$ being the set of all global variables and $V\mathcal{L}$ the set of all local variables. The global variables act as "classical" logical variables whereas a local variable may change its value from state to state. A special predicate ex is used to describe actions with ex(a) denoting the execution of action a. $\mathcal{T}_\Sigma$ is the set of all Σ-terms which is formed in the usual manner. The set $\mathcal{F}_\Sigma$ of well-formed Σ-formulae is defined by

Definition 1 (well-formed formulae) $\mathcal{F}_\Sigma$ is the smallest subset of $(\mathcal{V} \cup \Sigma \cup \{\neg, \wedge, \forall, \bigcirc, \Diamond, ;\})^*$ with the usual quantifier and connectives for negation and conjunction, and $\bigcirc\varphi \in \mathcal{F}_\Sigma$ ("next"), $\Diamond\,\varphi \in \mathcal{F}_\Sigma$ ("sometimes"), and $\varphi;\psi \in \mathcal{F}_\Sigma$ ("chop") for $\varphi, \psi \in \mathcal{F}_\Sigma$.

Notation: We use the usual reading for the abbreviations $\supset$ and $\equiv$.
From the defined operators others can be derived which are particularly useful in the context of planning and plan recognition in command language environments. $\Box\,\varphi$ is used as an equivalent to $\neg\Diamond\,\neg\varphi$. We have the following control structures:

$$\textbf{if } c \textbf{ then } \alpha \textbf{ else } \beta \equiv (c \supset \alpha) \wedge (\neg c \supset \beta) \quad \text{and}$$
$$\textbf{while } c \textbf{ do } \alpha \textbf{ od } ;\beta \equiv \textbf{ if } c \textbf{ then } (\alpha;\ \textbf{while } c \textbf{ do } \alpha \textbf{ od } ;\beta) \textbf{ else } \beta.$$

The assignment := is defined as a special action with $a{:=}b$ having the effect of giving a in the next state the current value of b.

Logical formulae are interpreted over intervals σ that are defined as non-empty sequences of states ($\sigma = (\sigma_0, \sigma_1, \ldots)$). The values of terms are determined by a Σ-interpretation $\mathcal{I}$ parameterized with a set of intervals W: Global variables are interpreted by an assignment $\beta : V\mathcal{G} \to D$ into domain D. The value of a local variable with respect to an interval $\sigma = (\sigma_0, \ldots \sigma_n)$ is its value in the initial state, i.e., states are assignment functions for the local variables. Additionally, they contain information about the command currently being executed. More complex terms are interpreted as usual with function symbols—just as predicate symbols—treated as globals.

The validity of a formula φ under the interpretation $\mathcal{I}$ is determined with respect to an interval $\sigma \in W$ (written $\sigma \models_{\mathcal{I}}$). We continue giving an informal description of the modal operators $\bigcirc, \Diamond$ and *chop*. A formal definition can be found in [BDK92]. $\bigcirc\varphi$ is said to be true in $\sigma = (\sigma_0, \sigma_1, \ldots)$ if φ is true in the interval beginning with the next state , i.e., in $(\sigma_1, \ldots)$. $\Diamond\,\varphi$ holds in σ if there is some suffix subinterval of σ in which φ holds. The chop operator ";" extends classical temporal logics as it provides a means of concatenating time intervals. We say $\varphi;\psi$ is true in $\sigma = (\sigma_0, \ldots, \sigma_{i-1}, \sigma_i, \sigma_{i+1}, \ldots)$ if σ can be split into two subintervals $\sigma_1 = (\sigma_0, \ldots, \sigma_{i-1}, \sigma_i)$ and $\sigma_2 = (\sigma_i, \sigma_{i+1}, \ldots)$ where φ is true in σ_1, and ψ is true in σ_2. We say that $\mathcal{I}$ is a *model* of φ if and only if $\sigma \models_{\mathcal{I}} \varphi$ for all $\sigma \in W$.

In the following sections the adoption and selection of hypotheses will be described in more detail.

4 Plan Recognition with Abductive Reasoning

As already outlined in section 2 there are cases of intuitively valid hypotheses which are not generated by "classical" abduction. To discuss the reasons in greater detail, we recall the definition of abduction.

Definition 2 (classical abductive explanation) Let $\mathcal{T}$ be a logical theory defined over the first-order language $\mathcal{L}$ and A a set of sentences of $\mathcal{L}$ called *abducibles*. φ is an *explanation* for ω if

(1) $\mathcal{T} \cup \varphi$ is consistent

(2) $\mathcal{T} \cup \varphi \models \omega$

(3) φ is a ground instance of some abducible $s \in A$.

The hypothesis $\Diamond a$ would not be accepted as an explanation for observation a, because condition (2)—the correctness criterion for abduction—is not fulfilled: $\Diamond a \not\models a$, for $\mathcal{T} = \emptyset$. This problem is overcome by using a weaker correctness condition. An explanation $\mathcal{P}$ is valid, if it can be refined with the observation ω. The refined explanations are determined by those models of $\mathcal{P}$ which are also models of ω (cf. also section 2). More formally we have

Definition 3 (refined explanations) Let $\mathcal{T}$ be a logical theory defined over LLP, $\mathcal{P}$ an LLP formula, and ω an observation. The set $\mathcal{C}$ of refined explanations is determined by

$$\mathcal{C}_{\mathcal{P},\omega} = \{Y \mid \mathcal{MOD}(\mathcal{T} \cup Y) = \mathcal{MOD}(\mathcal{T} \cup \mathcal{P}) \cap \mathcal{MOD}(\omega)\},$$

where $\mathcal{MOD}(\varphi)$ denotes the set of models of formula φ.

Consider again the hypothesis $\Diamond a$ and observation a. The refined hypotheses are determined by those models of $\Diamond a$ which are also models of a. These models can be split into those in which the action a occurs exactly at the first state and those in which it occurs in state one but also at some later point in time, which results in $\mathcal{C}_{\Diamond a,a} = \{a, a \wedge \bigcirc\Diamond a\}$. Note that a can be inferred from both formulae, i.e., the correctness is guaranteed . $\Diamond a$ is called the explanation for a.

Definition 4 (explanation) Let $\mathcal{T}$ be a logical theory defined over LLP, $\mathcal{P}$ an LLP formula, and ω an observation. $\mathcal{P}$ is an explanation for ω, if

(1) $\mathcal{T} \cup \mathcal{P}$ is consistent;

(2) For $\mathcal{P}$ there exists at least one refined explanation $\mathcal{P}'$ for which $\mathcal{T} \cup \mathcal{P}' \models \omega$;

(3) $\mathcal{P}$ is a ground instance—up to the current point in time—of some abducible.

Remarks:
Owing to a lack of space we are a bit careless with respect to point (3). An instantiation of the hypothesis is built by a special rule which uses a substitution obtained by the instantiations in the refined explanations.

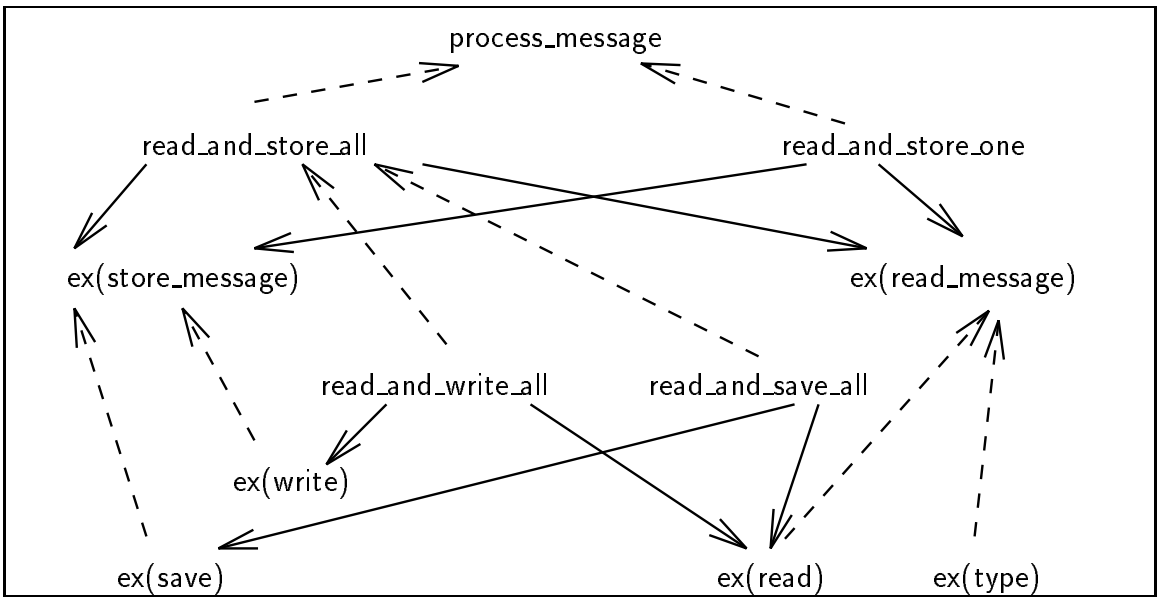

Figure 2: The plan hierarchy (PH_2)

If we have $\forall x(bird(x) \supset flies(x))$ and $flies(Tweety)$, then $bird(Tweety)$ is a valid explanation with respect to our definition as well, i.e., the explanations in the "classical sense" are also valid in our framework.
To determine the refined hypotheses we use an equivalence preserving transformation of LLP formulae into a specific graph syntax. The graph of formula $\mathcal{P}$ represents the models of $\mathcal{P}$ with nodes corresponding to interval states. Labelled arrows are used to determine the temporal order of states, e.g., the immediate successor (corresponding to the $\bigcirc$-operator) or the subsequence relationship (corresponding to $\Diamond$). Incorporating an observation into the graph means moving through it and filling nodes with the observation if this is feasible. Owing to a lack of space we are forced to omit a more detailed description. **Example:** Consider the example hierarchy PH_2 given below.[7]
Abstractions are formed as described in section 2. The plans contained in the decomposition hierarchy are

$$n := 1;\ \textbf{while}\ n < length(mbox)\ \textbf{do}\ ex(read\_message(n)); ex(store\_message(n)); n := n + 1\ \textbf{od} \quad (5)$$

$$n := 1;\ \textbf{while}\ n < length(mbox)\ \textbf{do}\ ex(read(n)); ex(write(n)); n := n + 1\ \textbf{od} \quad (6)$$

$$n := 1;\ \textbf{while}\ n < length(mbox)\ \textbf{do}\ ex(read(n)); ex(save(n)); n := n + 1\ \textbf{od} \quad (7)$$

$$ex(read\_message(x)); ex(store\_message(x)) \quad (8)$$

for which the decomposing plans are read_and_store_all (for (5)), read_and_write_all (for (6)), read_and_save_all (for (7)), and read_and_store_one (for (8)), respectively. Let all elements of the hierarchy be abducible. Suppose now that the user is seen to read message 1, i.e., we observe ex(read_message(1)). This observation fits into plan (5), as can be seen in the following. The while-plan is refined by first extracting the body of the loop for the current iteration according to the definition of while as a derived operator (cf. section 3), i.e., (5)

[7] which is quite similar to the Cooking Domain hierarchy of Kautz in [AKPT91], Chapter 2.

becomes

$$n := 1; \quad \begin{array}{ll} \textbf{if} & n < length(mbox) \\ \textbf{then} & ex(read\_message(n)); ex(store\_message(n)); n := 2; \\ & \textbf{while } n < length(mbox) \textbf{ do} \ldots \end{array} \tag{9}$$

Assuming $length(mbox) = 5$ the refined explanation is

$$ex(read\_message(1)); ex(store\_message(1)); n := 2; \ \textbf{while } n < 5 \textbf{ do} \ldots$$

because in those models of (9) in which ex(read_message(1)) is true we exactly expect ex(store_message(1)) as the next action and then the while-loop to be continued.

The problem is now how to find the explanations to test for refinement. We argue that for our application it is sufficient and more efficient to use heuristics to guide this selection. Thus, we will first of all test only those plans contained in the hierarchy which form a *decomposition*, in our example plans (5) to (8). For *abstract* plans and actions, e.g., read_and_store_all or read_message, we use the original stronger condition for explanations, i.e. $\mathcal{T} \cup \mathcal{P} \models \omega$, because the refinement step has already been made when adopting the "decomposed plans". These more abstract explanations are found by proving $\mathcal{T} \cup \mathcal{P} \models \omega$ with the aid of tactics. Thus, roughly said, to find these explanations we encode the formulae making up the plan hierarchy as sequent rules in a sequent calculus and the tactic is to choose only these rules when constructing hypotheses.[8]

These strategies allow us to produce only those plans which are of interest, i.e., contained in the hierarchy. This seems feasible to us, as otherwise the expressiveness of the language would give rise to a large set of explanations which would be of little use for plan recognition and be dismissed anyway, if the set of "best" explanations is determined. Examples are $\Diamond b, \Diamond c, \ldots$ as explanations for a.

The result of plan recognition gives rise to a set of possible explanations among which the best ones may be selected according to a probabilistic criterion (cf. section 5) or also according to specificity. In our help system context it may be desirable to have the most specific general explanation. This means we search for a plan as a hypothesis, not for an abstract action, but among the plans a specific one generally provides more information. For example, the while-plan (9) is more informative than the abstraction process_message.

Example (continued): Consider again the observation ex(read_message(1)). When plan recognition starts we search for explanations with valid refinements. These are (5), (6), (7), and (8). As we want to extract the *complete* set of feasible explanations, we also try to find those actions $\mathcal{P}$ which are classical abductive explanations, i.e., with $\mathcal{T} \cup \mathcal{P} \models \omega$. These are ex(read(1)) and ex(type(1)). For ex(read(1)) we can again find an explanation which has a refinement for this command, namely (5).

The set of abstract plans $\mathcal{P}$ for which we can prove $\mathcal{T} \cup \mathcal{P} \models \nu$ for all ν resulting from the previous steps is $V = \{$read_and_store_all, process_message, read_and_store_one(1)$\}$. Thus after the first recognition step the complete set of plan hypotheses is $V \cup \{(5), (6), (7), (8)\}$.

Observing in the next step the action ex(save(1)), we first test the refinements obtained so far for validity. This is not the case for (6) so this hypothesis is discarded and with it all dependent explanations as, e.g., read_and_write_all.

In the following section we will describe how to choose one hypothesis among the hypotheses obtained.

[8]The use of tactics for plan generation in LLP is discussed in [BDK92].

5 Selection of Plan Hypotheses

In this section, we will motivate the choice of Dempster-Shafer Theory, introduce its most important basic concepts, and show how they can be applied to the task of judging the quality of several hypotheses. Owing to a lack of space, we refer the reader to [Sha76, KSH91] for more details of DST.

There are two reasons for choosing DST instead of probability theory: First of all, DST (see [Sha76]) allows to work with underspecified models. This means that—in contrast to probability theory—it is not necessary to know *all* conditional and a priori probabilities or to introduce arbitrary independence assumptions in order to start computation. Instead, it is sufficient to give lower and upper bounds for the probabilities of some events and to leave the rest unspecified. This corresponds exactly to our situation: It is hardly possible to determine exactly the probability of a given plan hypothesis even on the basis of long-term observations of the user's behaviour—our knowledge of his preferences will always remain incomplete. DST enables such partial ignorance to be taken into account and to be distinguished from uncertainty, while probability theory requires the application of some meta-criterion like *maximal entropy* to artificially "complete" the given information.

The second reason is that the use of Dempster's rule for the combination of several pieces of evidence allows the process of narrowing the set of possible hypotheses to be adequately modeled when new observations are available as is the case for incremental plan recognition. With DST as the basic numerical formalism for dealing with uncertainty, we thus have a tool for modeling the initial situation in a granularity corresponding to our state of knowledge and the dynamic process of updating this description in the light of new evidence. In contrast, systems adopting probability theory—like Wimp3 (cf. [CG91]) which is based on dynamically created Bayesian networks— have to make numerous equiprobability assumptions in order to establish the required numerical values.

The basic idea of DST is that incoming information from a so-called *evidence space* induces a distribution of an *evidence mass* on the *hypothesis space*. This means that an observation makes us assign a certain degree of confidence to the various hypotheses. As time passes, new information will cause the evidence mass to be concentrated on a smaller number of hypotheses until eventually the correct one remains.

Now let Ω be the set of all single hypotheses, the so-called *frame of discernment*. We have then

Definition 5 (basic probability assignment, belief, plausibility) A mapping $m : 2^\Omega \to [0,1]$ is a *basic probability assignment* (bpa) iff

$$m(\emptyset) = 0 \quad \text{and} \quad \sum_{A \subseteq \Omega} m(A) = 1.$$

Given m, the functions Bel_m and $Pl_m : 2^\Omega \to [0,1]$, called *belief* and *plausibility*, resp., can be derived:

$$Bel_m(A) = \sum_{B \subseteq A} m(B), \quad Pl_m(A) = \sum_{B \cap A \neq \emptyset} m(B).$$

Wherever possible, the index m will be omitted.

From a logical point of view, a non-singleton subset A of Ω stands for the disjunction of all its members. While $m(A)$ represents the amount of confidence that can be attributed *exactly* to A (but owing to a lack of knowledge not to its subsets), $Bel_m(A)$ is the *total*

degree of confidence which can be assigned to A *and* its constituents, and $Pl_m(A)$ is that part of the evidence mass which might eventually be moved into A, given new information. That means, $Bel_m(A)$ and $Pl_m(A)$ form lower and upper bounds for the "true" but unknown probability of A. The numerical difference between both values is the *degree of ignorance* we have about A.

Remark: Using a bpa m_0 with $m_0(\Omega) = 1$ and $m_0(X) = 0$ for all proper subsets X of Ω, we can represent a state of *total ignorance* in which we only know that Ω contains the "true" value we are looking for. The corresponding belief function is called the *vacuous belief function.*

Two independent pieces of evidence—encoded in bpa's—can be combined using *Dempster's rule* the result of which is a new bpa which represents the information contained in both functions (cf. [Dem67]). The effect of applying this rule is that the evidence mass is concentrated on hypothesis sets being supported by both sources while diminishing the amount attributed to the others.

How can these notions be related to our needs in plan recognition? As the observations consist mainly of the user's actions, the set Cmd of all commands forms the *evidence space*, while the abstraction hierarchy PH_A of the plans is our *hypothesis space.* The fact that PH_A is typically not of the form 2^{Ω} has several implications which will be discussed below. We still lack a means of representing the *decomposition* of plans. We do so by introducing a mapping Δ between the evidence and hypothesis space:

$$\Delta : \ (2^{Cmd} - \emptyset) \rightarrow 2^{PH_A \times [0,1]} \tag{10}$$

which can be compactly written as a set of rules:[9]

$$\begin{array}{c} \textbf{if evidence } \omega \textbf{ then } H_1 \textbf{ with strength } s_{\omega,1} \\ \vdots \\ \textbf{then } H_n \textbf{ with strength } s_{\omega,n} \end{array}$$

for all $\omega \subseteq Cmd$, where $H_i \in PH_A$ and $s_{\omega,i} \in [0,1]$ such that $\sum_i s_{\omega,i} = 1$. Such a rule can be read as follows: If the (disjunctive) observation ω occurs and it "fits" into H_1 in the sense that H_1 can be refined with ω, then we assign to the (disjunctive) hypothesis H_1 an evidence mass of $s_{\omega,1}$ and so on for all H_i. If ω satisfies the structural requirements of all its consequences H_i, the application of such a rule induces a bpa s_ω on the hypothesis space via the various $s_{\omega,i}$. If, however, this connection between observation and hypothesis cannot be established in certain cases, the remaining $s_{\omega,i}$ have to be divided by an appropriate factor to ensure the properties of a bpa. Causes of such a (partial) failure of a rule include incompatibilities between the actual parameters of ω and the variable bindings in H_i or a conflict because of the temporal structure of H_i (that means, ω was expected for another point in time).

The selection phase of the plan recognition process now works as follows: Assume we are given a bpa m_0 over the set of plan hypotheses which represents either an initial valuation in the sense of a user model or the result of the last plan recognition step. As a new observation ω occurs, the corresponding rule is applied using information about its applicability provided by the abductive recognition module which yields a new bpa s_ω

[9] Δ actually is a generalization of the multivalued mapping Γ from [Dem67] which—given a *sensor space* with a probability function—induces a bpa on the hypothesis space. The foundations for this extension can be found in [Yen86, Yen89, GB91].

over the plan hypotheses. m_0 and s_ω are combined with Dempster's rule and the resulting bpa m_1 now mirrors the impact of the recent observation on the numerical valuation of the hypotheses. This result can now be used to define various selection criteria as will be demonstrated in the following example.

Example (continued): Assume that after the first observation ex(read_message(1)), we have the bpa m_1 which assigns 0.2 to {read_and_write_all}, 0.5 to read_and_store_all, 0.1 to {read_and_store_one},and 0.2 to process_message. Here, read_and_store_all stands for the set { read_and_write_all, read_and_save_all } of hypotheses being subsumed by it in the plan hierarchy depicted in figure 2. Accordingly, process_message acts as the name of the set of all single hypotheses. This bpa tells us that the user obviously tends to process the whole contents of his mailbox at once, because the probability for the most general hypothesis describing this behaviour (read_and_store_all) lies between 0.7 and 0.9, whereas read_and_store_one ranges between 0.1 and 0.3. The second observation ex(save(1)) triggers the following rule

if evidence save **then**	{read_and_save_all}	**with strength** 0.4
	read_and_store_all	**with strength** 0.3
	{read_and_store_one}	**with strength** 0.3

which is completely applicable and—after combination with m_1 using Dempster's rule—yields the result

$$\text{read\_and\_store\_all} - 0.362, \{\text{read\_and\_save\_all}\} - 0.483, \{\text{read\_and\_store\_one}\} - 0.155.$$

This means that our belief in the user pursuing one of the plans concerning the whole mailbox has grown to 0.845, with the variant using the save command being preferred over the one applying the write command. If the system is now forced to opt for one single plan, the output will be read_and_save_all because this is the one with the highest valuation: Our belief in it is 0.483. Another selection criterion might be the highest plausibility value among the single hypotheses, which—in this case—would yield the same result.

Remarks: As mentioned above, the idea of considering only a part of the whole power set of all hypotheses has several implications (cf. [GS85]): In this case, bpas can be combined in polynomial time (instead of exponential time in the general case), but this combination is no longer associative, i.e., the order of combination may influence the result. In addition, the plausibility of a hypothesis A no longer corresponds to $1 - Bel(\overline{A})$[10] if $\overline{A}$ does not belong to that part of the hypothesis space currently being considered, thus violating a basic equation from DST.

6 Conclusion and Outlook

In the preceding sections we have presented a new concept for plan recognition based on a generalized approach to abduction in the modal logic LLP. A Dempster-Shafer based selection module serves to determine the most plausible hypotheses at any time.

The first formal theory of plan recognition based on deductive inferences in a closed plan hierarchy (cf. also section 2) was developed by Kautz (see [Kau87], [KA88]). Recent work in plan recognition focuses on the problems not solved by this approach (compare

[10]Here, $\overline{A}$ stands for the complement of A.

[GL92]). For example, Appelt and Pollack (cf. [AP90]) use the concept of *weighted abduction* (also see [HSME89]) to allow the indeterministic choice of a single plan or the determination of more likely plans to prevent overcommitment, i.e., the premature selection of one plan, if required. However as Goodman and Litman state (see [GL92]) all these works are of a more theoretical importance as the algorithms developed are not adapted to the actual use of plan recognition in intelligent systems. In contrast, we provided evidence for the fact that our formal framework for plan recognition possesses practical applications in realistic scenarios, because—among others—it allows plans containing control structures such as loops and conditionals to be recognized. As the logic LLP is also well suited to planning tasks (cf. [BDK92]), a uniform framework for both planning and plan recognition tasks is obtained (see also [AKPT91]).

Concerning the handling of uncertainty, we take into account a priori probabilities of the hypotheses and keep the numerical computations for the next recognition step as the basis of decision-making instead of heuristics. This is in contrast to Carberry's approach in [Car90a].

The concepts described above are currently being implemented within the project PHI (Plan-based Help Systems) at the German Research Center for Artificial Intelligence (see also [BBD+93]).

Acknowledgments: We would like to thank Susanne Biundo for helpful comments on an earlier version of this paper.

References

[AAA86] *Proceedings of the 5th National Conference of the American Association for Artificial Intelligence*, Philadelphia, PA, August 1986.

[AKPT91] J.F. Allen, H.A. Kautz, R.N. Pelavin, and J.D. Tenenberg. *Reasoning About Plans*. Morgan Kaufmann, Los Altos, CA, 1991.

[All79] J.F. Allen. A plan-based approach to speech act recognition. Technical Report (Doctoral Dissertation) 131/79, University of Toronto, Department of Computer Science, 1979.

[AP90] D.E. Appelt and M. Pollack. Weighted abduction for plan ascription. Technical report, Artificial Intelligence Center and Center for the Study of Language and Information, SRI International, Menlo Park, California, 1990.

[BBD+93] M. Bauer, S. Biundo, D. Dengler, J. Köhler, and G. Paul. PHI—a logic-based tool for intelligent help systems. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, Chambéry, France, August 1993. Morgan Kaufmann Publishers.

[BDK92] S. Biundo, D. Dengler, and J. Köhler. Deductive planning and plan reuse in a command language environment. In *Proceedings of the 10th European Conference on Artificial Intelligence*, pages 628–632, 1992.

[BN92] H.-J. Bürckert and W. Nutt. On abduction and answer generation through constrained resolution. Research Report RR-92-51, DFKI, 1992.

[Car90a] S. Carberry. A model of plan recognition that facilitates default inferences. In *Proceedings of the Second International Workshop on User Modeling (UM90)*, Honolulu, Hawaii, 1990.

[Car90b] S. Carberry. *Plan Recognition in Natural Language Dialogue.* MIT Press, Cambridge, MA, 1990.

[CG91] E. Charniak and R. Goldman. A probabilistic model of plan recognition. In *Proceedings of the 9th National Conference of the American Association for Artificial Intelligence*, pages 160–165, Anaheim, CA, July 1991. MIT Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts.

[Con86] R.L. Constable. *Implementing Mathematics with the Nuprl Proof Development System.* Prentice-Hall, 1986.

[Dem67] A.P. Dempster. Upper and lower probabilities induced by a multivalued mapping. *Annals of Mathematical Statistics*, 38:325–339, 1967.

[GB91] J. Guan and D. Bell. *Evidence Theory and its Applications*, volume 1. Elsevier, 1991.

[GL92] B. Goodman and D. Litman. On the interaction between plan recognition and intelligent interfaces. *User Modeling and User-Adapted Interaction*, 2(1-2):83–116, 1992.

[Gou50] Th.A. Goudge. *The Thought of C.S. Peirce.* Dover Publications Inc., New York, 1950.

[GS85] J. Gordon and E. Shortliffe. A method for managing evidential reasoning in a hierarchical hypothesis space. *Artificial Intelligence*, 26:323–357, 1985.

[HRS90] M. Heisel, W. Reif, and W. Stephan. Tactical Theorem Proving in Program Verification. In *Proceedings of the 10th Conference on Automated Deduction*, volume 449 of *LNCS*, pages 117–131. Springer, 1990.

[HSME89] J. R. Hobbs, M. Stickel, P. Martin, and D. Edwards. Interpretation as abduction. Draft version, SRI International, Artificial Intelligence Center, Menlo Park, CA, 1989.

[KA88] H. Kautz and J. Allen. Generalized plan recognition. In AAAI-86 [AAA86], pages 32–37.

[Kau87] H. Kautz. *A formal theory of plan recognition.* PhD thesis, University of Rochester, 1987.

[KKT92] A.C. Kakas, R.A. Kowalski, and F. Toni. Abductive logic programming. Draft version, Department of Computer Science, University of Cyprus, Nicosia, and Imperial College of Science, Technology and Medicine, London, 1992.

[Krö87] F. Kröger. *Temporal Logic of Programs.* Springer, Heidelberg, 1987.

[KSH91] R. Kruse, E. Schwecke, and J. Heinsohn. *Uncertainty and Vagueness in Knowledge-Based Systems.* Springer-Verlag, Berlin, Heidelberg, New York, 1991.

[May92] J. Mayfield. Controlling inference in plan recognition. *User Modeling and User-Adapted Interaction*, 2(1-2):55–82, 1992.

[Pau93] G. Paul. Approaches to abductive reasoning—an overview. *Artificial Intelligence Review*, 7:109–152, 1993.

[Pei58] C.S. Peirce. *Collected Papers of Charles Sanders Peirce (eds. C. Hartshorne et al.).* Harvard University Press, 1931-1958.

[RP86] R. Rosner and A. Pnueli. A choppy logic. In *Symposium on Logic in Computer Science, Cambridge, Massachusetts*, pages 306–313, 1986.

[Sha76] G. Shafer. *A Mathematical Theory of Evidence.* Princeton University Press, Princeton, 1976.

[Wil78] R. Wilensky. Understanding goal-based stories. Research Report (Doctoral Dissertation) 140, Yale University, Computer Science Department, 1978.

[Yen86] J. Yen. A Reasoning Model Based on an Extended Dempster-Shafer Theory. In AAAI-86 [AAA86], pages 32–37.

[Yen89] J. Yen. GERTIS: A Dempster-Shafer Approach to Diagnosing Hierarchical Hypotheses. *Communications of the ACM*, 32(5):573–585, 1989.