Non ridere, non lugere, neque detestari, sed intelligere!
(Not to laugh, not to lament, nor to detest, but to understand!)

Benedict de Spinoza(1632-1677), "Tractatus Politicus"

University of Alberta

Describing plan recognition as non-monotonic reasoning and belief revision

by

Pawel Jachowicz ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science.

Department of Computing Science

Edmonton, Alberta
Spring 1998

0-612-28949-4

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Describing plan recognition as non-monotonic reasoning and belief revision** submitted by Pawel Jachowicz in partial fulfillment of the requirements for the degree of Master of Science.

Dr. Randy G. Goebel

Dr. Peter van Beek

Dr. Oliver Schulte

Dr. L.Y. Yuan

Date: Mar. 6/98

This thesis is dedicated to my parents.

Abstract

We provide a characterization of plan recognition in terms of a general framework of belief revision and non-monotonic reasoning. We adopt a generalization of classical belief revision to describe a competence model of plan recognition which supports dynamic change to all aspects of a plan recognition knowledge base, including background knowledge, action descriptions and their relationship to named plans, and accumulating sets of observations on agent actions.

Our plan recognition model exploits the underlying belief revision model to assimilate observations, and answer queries about an agent's intended plans and actions. Supporting belief states are determined by observed actions and non-monotonic assumptions consistent with background knowledge and action descriptions.

We use a situation calculus notation to describe plans and actions, together with a small repertoire of meta predicates which are used to specify observations to the belief revision system, and to query the reasoning system regarding the current status of plans and predictable actions.

Our intent is to demonstrate the connections between a general plan recognition model and important concepts of belief revision and non-monotonic reasoning, to help establish a basis for improving the specification and development of specialized plan recognition systems.

Acknowledgements

My thanks to the examining committee of Dr. Randy Goebel, Dr. Peter van Beek, Dr. Oliver Schulte and Dr. Li-Yan Yuan for their comments and suggestions.

My heartfelt thanks to my supervisor Professor Randy Goebel for believing in me and for giving me the opportunity to pursue this research. His guidance during my stay in Edmonton and after my moving to Calgary allowed me to complete this research. Thank you Randy for spending so many hours during the weekends to meet with me and to discuss this work. Thank you also for your financial support during the course of this research, without which it may have been impossible to complete it. Thank you for your friendship.

I would especially like to thank my parents for all of their emotional support during all of my years of study at the University of Alberta. I am the luckiest person in the world to have parents like you. Thank you from the bottom of my heart for your love and understanding.

Sincere thanks to my friends Wladek and Magda for always being there and for being so supportive. Thanks to Peter and Sylvia for all the good and bad times which we have had at this University. Special thanks to Malgosia, Kasia and Ania for playing an important role in my life during the course of my studies and for remaining wonderful friends. Also thanks to Dimos, Marcus, Lester, Nicholas and Andrzej for the fun times.

Contents

List of Figures

Chapter 1

Introduction

1.1 Problem specification

Knowing the plan an agent is pursuing is important for several reasons. It allows us to predict actions the agent might take in the future, and it allows us to aid or hamper the agent by suggesting or even taking action alternatives.

In its simplest conception, a plan explains a sequence of actions if they comprise the plan. To recognize an observed set of actions as a plan first requires that one establish a representation of plans as a named or similarly identified set or sequence of actions. With that plan representation and given a set of observed actions, a plan recognition system constructs the set of possible plans which explain the specified actions [9].

Like all recognition tasks, the object to be recognized has to be described in terms of some number of components. Sentences are sequences of words, programs are sequences of instructions, and plans might simply be conceived as named sequences of actions. The concept of "sequence" is typically too simple however, and can be elaborated along at least two dimensions. First, a plan of any practical complexity will likely include alternative actions to accomplish the same subgoals (which turns any plan description into a tree or lattice). Second, any set of actions to accomplish any particular plan will typically have optional actions which serve only to embellish the goal; so there will be necessary and contingent actions for any practical plan.

Within this kind of situation, a plan recognition system must be able to use incomplete information to provide the required flexibility. A general plan recognition system must be able to perpetually accept revised descriptions of a world in which actions take place, including changes to the observed relationships amongst actions and plans. To anticipate the behaviour of agents acting in a dynamic observable world, the plan recognition system must be able to hypothesize consistent plans. Within this kind of framework, it is unsurprising that non-monotonic reasoning and belief revision will provide a basis for reasoning in this kind of incomplete information context.

At least four different general plan recognition strategies exist in the literature: parsing [21], plausible inference [2] and circumscribing a hierarchical representation of plans using deduction [9] and abduction [12].

The method proposed here uses ideas related to at least the last three of these methods, developed within a framework for managing and reasoning about beliefs. We use the Ghose-Goebel belief change model to maintain a dynamic set of assertions that represent actions and relationships amongst actions and plans. This method explicitly supports reasoning with incomplete knowledge and revision of the belief base with newly observed facts. Observed actions are assimilated within the belief revision system and, together with current beliefs about actions and their affects, constrain the plans that can be recognized. Additionally, The Ghose-Goebel (henceforth GG) reasoning framework supports the maintenance of multiple mutually inconsistent states of the world, providing a basis to assume alternative hypothetical completions of plans.

1.2 Motivation

One of the central issues of Artificial Intelligence deals with representing plans and actions. Plan recognition allows for reasoning with observed or completed actions in terms of constructing plans which contain these actions. This type of reasoning is very important and useful in the areas of story understanding, strategic planning, and scheduling.

In order for a plan recognition system to be really useful, it should interact with the changing environment by dealing with uncertainties inherent to a dynamic plan library. The use of belief revision is an attempt to incorporate possible uncertainty handling to the plan recognition system. For example, by finding the set of assumptions (non-monotonically) that entail the action observations, we are able to hypothesize the observed agent's goals according to the plans contained within the plan library. The many possible consistent sets of assumptions entailing the observations may be handled by the GG belief revision reasoning framework.

Current techniques used for solving the plan recognition problem deal with static knowledge bases, where a plan library already exists and the relationships amongst its comprising actions are known. In such systems, certain heuristics are applied to recognize viable plans. Carberry's focusing heuristic [2] hypothesizes a set of viable plans, selects the "best" one and incorporates it into a context model. Kautz [9] presents yet another heuristic used for selecting plans amongst multiple plausible plans.

Our method does not pretend to know which of the possible plans might be deemed as "the best." Instead, the GG belief revision framework allows the maintenance of all feasible plans which are recognizable, based on given observations. Eventually, and in any practical situation, such preferences for particular plans will be captured as epistemic entrenchment conditions [14]. We claim it is unnecessary and in fact dangerous to make a premature commitment to some plausible plan, when in fact all the possible plans ought to be presented to reflect the competence of a plan recognition system. This is consistent with the belief revision principle of informational economy which states that belief states should change in such a way as to maintain the maximal amount of information from state to state [5].

By using the techniques developed in belief revision, we explicitly allow for our knowledge base to consist of incomplete knowledge. We incorporate new observations into our knowledge base using the belief revision expansion, revision and contraction operators [5, 6, 15]. Our plan library is therefore dynamic. This mimics a more natural temporal varying behaviour of beliefs: in every reasonable possible world changes occur. As observations are made, the plausibly inferred plans are incrementally determined by belief states.

1.3 Scope of this thesis

We provide a new way of looking at the plan recognition problem. Not only do we expand the definition of plan recognition to include the possibility of having dynamically changing plan libraries, but also we describe a way of handling such environments. We propose the use of a belief revision framework known as the Ghose-Goebel framework which is capable of dealing with the dynamically changing nature of plan libraries to maintain the changing knowledge about the world. We make use of non-monotonic reasoning to make certain assumptions which aid us in recognizing plans. This dissertation contains the following sections:

- Description of the existing methods currently used to solve the plan recognition problem. Among these are Henry Kautz' [9] monumental plan recognition work, Marc Vilain's [21] use of parsing context-free grammars, Charniak and Goldman's [3] probabilistic approach to deal with plan recognition via Bayesian networks and other approaches.

- Description of the GG reasoning framework, which is used in the belief maintenance module to keep track of how new observations affect the existing knowledge base, and what to do with inconsistencies. Included is a comparison of this model with the traditional AGM belief revision model and reasons for selecting the GG model over the traditional one.

- Description of plan recognition based on the belief maintenance module using non monotonic assumptions to recognize and predict possible plans and actions which may potentially occur.

- Knowledge representation of axioms (including actions, plans and state affecting axioms) which support the plan recognition model.

- The evaluation of the benefits and drawbacks of using this plan recognition model.

- Implementation design characteristics and performance issues of such a model are examined.

- Future work which may be performed to further develop this *dynamic* method of solving the plan recognition problem.

1.4 An overview of the system

The system we propose consists of two modules. The belief maintenance module uses the Ghose-Goebel Belief Revision System to maintain a body of knowledge based on observations. This system maintains a plan library and a knowledge base. It allows for dynamic change of both of these based on new information observed by an agent. The knowledge base or the plan library may be expanded, revised or contracted, depending on the type of new information which is observed. Multiple mutually inconsistent belief sets are maintained as potential belief states that can be used to recognize plans.

The second module is responsible for plan recognition using the information about the world maintained by the first module. This system allows for observation of new information, certain types of predictions, and recognition of possible plans in the given state of the world. It relies on non monotonic assumptions which are used in the plan recognition and prediction aspects of its functionality.

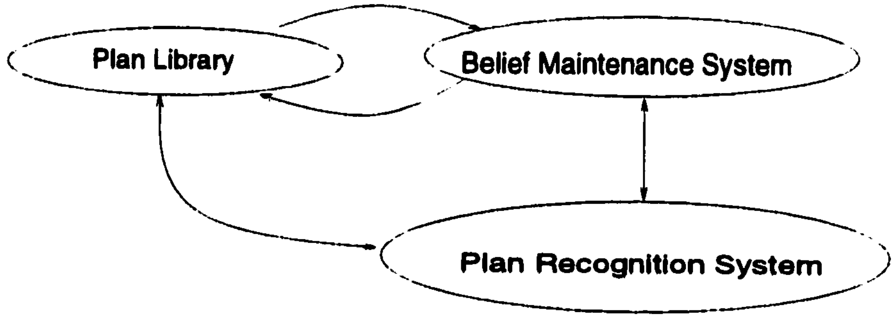


Figure 1.1: Overview of our system.

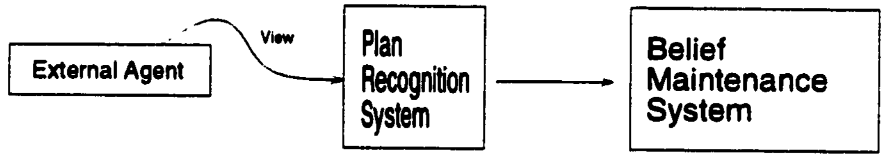


Figure 1.2: The system as seen by the agent.

Figure 1.1 shows the two modules interacting with the plan library. The belief maintenance system maintains its current status, while the plan recognition module uses it to predict and recognize possible plans and actions.

When viewed from the outside, the belief maintenance module is completely transparent. It is not important to the agent to know the intricacies of maintaining a valid knowledge base and plan library. What the agent is interested in is the plan recognition system because this system actually provides to the agent the information the agent requires in the form of predictions and plan recognition. Furthermore, any observations that the agent makes are passed on to the belief maintenance module which appropriately revises the knowledge base or the plan library as to maintain informational economy and consistency among the many potential states of the world which may result.

Figure 1.2 shows a graphical representation of the plan recognition module seen by the agent. This module makes continuous uses of the belief maintenance module which keeps track of all revisions based on observations made by the agent.

1.5 Assumptions

We base our system on a set of assumptions which define guidelines for comparison with other existing systems. These assumptions also provide an outline as to what the system can be expected to do and what its limitations are. Some of these assumptions could be loosened, but this would impact the competency of representing plan recognition by means of belief revision.

1. All knowledge about the world is contained in the two repositories labeled *plan library* and *knowledge base*. The *plan library* is responsible for holding all information which deals with representing known plans. These are represented by sequences of actions. The *knowledge base* contains all other information about the world, as required by the reasoning engine to draw conclusions based on the plan library. The information here includes different types of axioms which describe potential states, effect axioms of actions, frame axioms, as well as other types of axioms which are required for reasoning.

2. We leave it up to the reasoning engine underlying the belief revision system to figure out how to use the information in the plan library and the knowledge base.

3. We do not require the belief maintenance system to make choices between potential belief states which are created when the plan library or knowledge base are revised by newly observed information. In fact, we adopt the Ghose-Goebel Belief Revision model whose inherent property is the ability to maintain all possible potential belief states, since each one has a likelihood of being correct. We do not know whether this approach will require copious amounts of storage and time to be practically implementable, but our objective is not optimality but competency of representing plan recognition with belief revision.

4. We are guided by the principle of informational economy which states that no information should be discarded, however unlikely it may appear, if there is the

slightest chance that it in fact may be true in some future state.

5. We allow nonmonotonic assumptions to guide our plan recognition process. We assume that if no evidence exists which would preclude a fact or action from holding or being possible, then such fact or action may be used in the plan recognition process. This assumption is important because it gives us a freer hand in recognizing plans. If no evidence exists which would be inconsistent with recognizing a plan, that plan should be feasible, even if supporting evidence for it has not yet been observed.

6. We leave it up to various epistemic entrenchment techniques to support any selection of *best* plans. The value of the entrenchment of various facts, axioms, plans or other notions may be based on multiple factors depending on the situation. Some measures of entrenchment have been based on probability, utility and source of information; in most instances, previous plan preference methods fit easily into this framework.

Chapter 2

Belief maintenance system

2.1 Introduction

Whenever beliefs change, inconsistencies may result. These inconsistencies are caused by relationships between facts and rules which make up our beliefs. Since these relationships are very often complex, any change to any of the facts and rules causes a cascade of changes which have to propagated in order to maintain consistency.

A belief maintenance system should provide a comprehensive competence theory for the process of belief change. It should allow for new information to be added and subtracted from a knowledge base describing a situation of the world. It should also allow for information in the knowledge base to be modified without being added or subtracted. This modification is commonly referred to as *belief revision.*

2.2 Definitions

Before discussing further details, we informally describe some of the key components of our chosen representation. We use a version of situation calculus syntax based roughly on work by Lifschitz [11] and Kowalski [10].

We initially distinguish of three predicates: *holds, observed* and *goal.* The predicate *holds* allows us to describe each state in terms of fluents which are claimed to

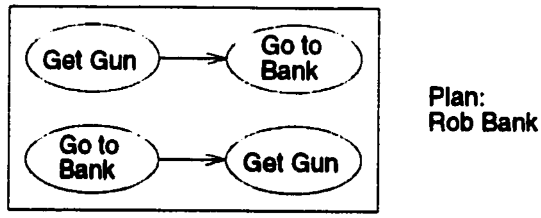


Figure 2.1: The plan RobBank consisting of two actions.

be true in a state.

A *knowledge base* is a collection of axioms. These axioms describe actions, action effects, and relationships amongst fluents, actions, and plans. A *belief state* is a situation represented by constant symbols, e.g., s_0 or by composite terms $result(a, s_0)$ when a denotes an action. A *fluent* is a truth function defined on a state:

$$holds(hasGun(Fred), s_0) \tag{2.1}$$

is a fluent denoting the fact that Fred has a gun in state s_0, while

$$\neg holds(isInBank(Fred), s_2) \tag{2.2}$$

is a fluent denoting the fact that in state s_2 Fred is not in the bank. An *action* is an n-ary function from states to states, e.g., as in the agent named "Fred" in the action $getGun$, here asserted to have taken place in state s_1:

$$holds(hasGun(Fred), result(getGun(Fred), s_1)) \tag{2.3}$$

The simplest form of a *plan* is a sequence of actions leading from some state to some subsequent state. Actions are contained in a plan if a plan explains them: e.g.,

$$goal(toHunt, s_0) \leftarrow predictable(getGun(Fred), s_0) \tag{2.4}$$

This simple one action plan "to hunt" is one explanation for the "get a gun" action [11]. If it is possible for Fred to obtain a gun or if at least we can assume this, then also it is possible to satisfy the goal of hunting.

The *holds* relation also allows us to describe relations amongst actions and fluents by specifying state axioms, effect axioms and frame axioms. The *observed* relation serves the purpose of distinguishing actions which we have not "seen" an agent change from actions which have caused fluents to be included into our knowledge base. Example 2.5 shows how a fluent observed in some state holds in the next state.

$$
\begin{aligned}
&holds(isLoaded, s_1) \Leftarrow \\
&observeFluent(isLoaded, s_0) \wedge s_0 \leq s_1
\end{aligned}
\tag{2.5}
$$

Similarly the trivial example 2.6 demonstrates the relationship between observing an action and knowing that an action is possible. If an action is observed in some state, than it is executable in that state. For precise definitions of executable, assumable and predictable actions refer to section 4.2.

$$
\begin{aligned}
&possible(loadGun(X), s_0) \Leftarrow \\
&observeAction(loadGun(X), s_0) \wedge s_0 \leq s_1
\end{aligned}
\tag{2.6}
$$

Finally, the predicate *goal* allows us to define a relationship between plan names and a situation, and can be viewed as a plan recognition-specific instance of a non-monotonic derivability relation like that used, for example, in hypothetical reasoning systems like Theorist [16, 17]. Example 2.7 expresses the relation that it is plausible that one has the goal *robBank*, in the situation resulting from the actions of getting a gun and going to a bank. The predicate *predictable* determines whether:

1. The contained action can be executed based on the information returned by the plan recognition module.

2. The contained action can be non-monotonically assumed to be executable by that nothing contradicts it.

$$
\begin{aligned}
&goal(robBank(X), S) \Leftarrow \\
&predictable(goToBank(X), S) \wedge \\
&predictable(getGun(X), S)
\end{aligned}
\tag{2.7}
$$

See section 4.2 for analysis of what predictability, possibility and assumability relations may be defined on actions and plans.

2.2.1 Basic operators

Belief revision is the process of changing a belief state according to some new information. We use three operators to define belief state change:

Expansion (equation 2.8) defines how to add new information to the knowledge base. This might be a new observation, the definition of an action description, some new relationship between actions and their effects, or even a new plan. The idea of expansion is to assume that one can simply append the knowledge base with new information which does not result in any portion of the knowledge base becoming inconsistent.

If our belief state is represented by a theory K, and K is expanded by a fact x, commonly written K_x^+, then, under the AGM theory, there is formed a union between the consequences of K and the new fact x.

$$Cn(K)_x^+ = Cn(Cn(K) \cup \{x\}) = Cn(K \cup \{x\}) \tag{2.8}$$

Contraction (equation 2.9, denoted by K^-) defines the process of removing a fact x from theory K in such a way as to eliminate from K the possibility of believing x. Therefore if x is accepted in K_x^-, it ought also be accepted in the intersection of K and K_x^* (which is the revision operator described next). The *Harper identity* is based on this observation, and defines contraction in terms of revision. The idea is to try to eliminate from the knowledge base some fact which we see being invalidated by observation. If no evidence points towards the removal of a fact after its addition to a theory, such a fact ought not to be abandoned:

$$Cn(K)_x^- = Cn(Cn(K) \cap Cn(K_x^*)) \tag{2.9}$$

Revision (equation 2.10), denoted as K^*, defines how to add a potentially inconsistent fact x to a theory K, as long as the new theory K_x^* is consistent and closed under logical consequence [15]. Abstractly, the AGM version of revision anticipates the need to remove some potentially conflicting information, and so stipulates that "minimal changes" ought to be made to the theory which is being revised. This desire is motivated by an equally abstract goal of preserving the postulate of informational economy.

The *Levi identity* may be employed to define revision in terms of contraction and expansion. It is important to keep in mind that revision will most likely result in multiple logical theories within our knowledge base, because there will most likely be multiple ways of contracting x.

$$Cn(K)^*_x = Cn((K^-_{\neg x}) \cup \{x\}) \tag{2.10}$$

2.3 Classical belief revision

Classical belief revision usually refers to the Alchourrón-Makinson-Gärdenfors (AGM) model of belief revision [5, 6, 15]. Their systematic study of belief change consists of [6]:

1. A specification of expressible beliefs which an agent may have.
2. A way for representing a belief.
3. A specification of operators to update a belief state with new beliefs which will guide belief change.
4. A set of rationality postulates which specify constraints on belief change operations.

The AGM model uses the definition of a knowledge base K, which is a deductively closed propositional theory, and describes any given state. Various beliefs are valid, false or unknown.

1. If belief x is valid, then $K \models x$.

2. If belief x is false, then $K \models \neg x$.

3. If belief x is unknown, then $K \models x$ or $K \models \neg x$.

Based on these possibilities, the AGM model makes use of the operators discussed in Section 2.2.1. Furthermore, for each operator a set of guiding postulates is presented which constrain the operator and may be used to test candidate resulting belief states. Here are the postulates for *contraction*:

1. K_x^- is a theory.

   This is the *closure* postulate which states that after performing the contraction operation, a theory should result.

2. $K_x^- \subseteq K$.

   The *inclusion* postulate states that the revised theory is contained in the old one.

3. $If \;\; x \notin K$, then $K_x^- = K$.

   The *vacuity* postulate states that if contraction is attempted on a belief which does not hold, then no changes should be made to the knowledge base.

4. $If \;\; x \notin Cn(\emptyset)$, then $x \notin K_x^-$.

   The *success* postulate says that contraction will remove any belief, unless it is a tautology.

5. $If \;\; Cn(x) = Cn(y)$, then $K_x^- = K_y^-$.

   The *preservation* postulate ensures that the syntax of the contracted belief should not impact the resulting theory.

6. $K \subseteq Cn((K_x^-) \cup x)$.

   The *recovery* postulate states that if a belief is removed and then added back to the knowledge base, sufficient information in the knowledge base should remain to arrive at the original state of the knowledge base, prior to contraction. The original belief is recovered.

7. $(K_x^-) \cap (K_y^-) \subseteq K_{x\wedge y}^-$.

   The first supplementary postulate states that if a conjunction of beliefs is removed, then the resulting belief state should have more information than if the beliefs were contracted individually.

8. If $x \notin K_{x\wedge y}^-$, then $K_{x\wedge y}^- \subseteq K_x^-$.

   The second supplementary postulate states that if a belief is not in the belief base resulting from a contraction of a conjunction of beliefs, then the belief base resulting from such a conjunction contraction is a subset of a belief base which results after contracting just a single belief.

2.4 Challenges for the AGM model

In order for a belief revision theory to be competent, to explain what is meant by *ideal belief change*, some difficulties of the AGM theory of belief revision must be presented. As pointed out by [6], among the most serious of these are:

1. An inadequate method of providing for retraction of beliefs. The retraction of a belief is simply not recorded in the new belief state. This contrasts with the fact that the addition of a belief is recorded.

   For example, let K_0, the initial belief state be

   $$K_0 = Cn(\{a \rightarrow b\}) \tag{2.11}$$

   Here a and b are arbitrary logical sentences such as:

   $a =$ *Tweety is a bird.*

   $b =$ *Tweety flies.*

Now if we contract K_0 with b, we get:

$$K_1 = (K_0)^-_b = Cn(\{a \rightarrow b\}) \tag{2.12}$$

Contraction is so defined that K_1 is represented identically to K_0 because b is not a consequence of the beliefs contained in K_0. Now if we expand the beliefs in K_1 with a, we obtain

$$K_2 = (K_1)^+_a = Cn(\{a, a \rightarrow b\}) \tag{2.13}$$

In belief state K_2, b is a valid belief. The fact of contracting b from K_0 has not been recorded, and b has been unintentionally added to the new belief state K_2, despite its earlier direct contraction.

2. The AGM belief revision framework does not specify a method for revising a belief beyond a single step. This is clearly demonstrated by the example just provided. The contraction of b has not been recorded and the subsequent step has revised the belief state is such a way, that the information contracted is held to be true.

3. Similarly, the principle of informational economy is not preserved in the AGM theory of belief change. This principle provides a guiding strategy for performing belief revision and states that, whenever possible, information ought not to be needlessly discarded.

   In the example just provided, if the belief state K_2 is revised with b, then there are three possible potential resulting belief states:

$$Cn(\{a, a \rightarrow b\}) \tag{2.14}$$

$$Cn(\{a\}) \tag{2.15}$$

$$Cn(\{a \rightarrow b\}) \tag{2.16}$$

The AGM theory forces the selection of one of the three states and discards the others. This approach assumes the existence of certain priority orderings amongst the sentences. Given this ordering (epistemic entrenchment) [14] (sections 2.6 and 4.5.1), the AGM theory selects one of the orderings arbitrarily. However, [6] argues that without explicit ordering information, all three must be maintained until some new information becomes known, which allows us to choose the appropriate ordering. Otherwise, a potentially valid belief state may be lost.

4. Similarly, the AGM theory does not explain how beliefs should change if credibility of information is unknown. This allows for information to be discarded prematurely, when in fact the credibility may become known at some future time.

2.5 The Ghose-Goebel model of belief revision

We use the Ghose-Goebel belief change model to maintain a dynamic set of assertions that represent actions and relationships amongst actions and plans. This method explicitly supports reasoning with incomplete knowledge and for revision of the belief base with newly observed facts without unnecessarily discarding potential beliefs. Observed actions are assimilated within the belief revision system and, together with current beliefs about actions and their affects, constrain the plans that can be recognized. The Ghose-Goebel reasoning framework supports the maintenance of multiple mutually inconsistent states of the world, providing a basis to assume alternative hypothetical completions of plans.

2.5.1 Expansion

To provide a simple concrete example, we provide some details based, again, on the style of Kowalski. Under that scheme, state description axioms (Equation 2.17) are used to describe the status of the fluents of a potential state. This information is used to determine which actions are possible in a state. The axioms here are roughly based

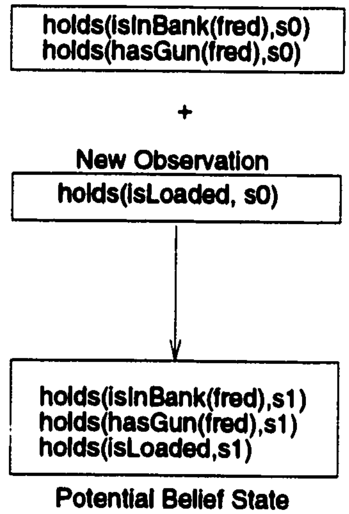


Figure 2.2: Expansion of a knowledge base with a state description axiom.

on the problem discussed by Hanks and McDermott [8]. In this case, two fluents are true: Fred is in the bank and has a gun. The original knowledge base consists of:

$$holds(isInBank(fred), s_0) \qquad (2.17)$$
$$holds(hasGun(barney), s_0)$$

When the knowledge base in Equation 2.17 is expanded with the axiom 2.18, the resulting knowledge base will have this axiom included. The use of the expansion operation presumes that the addition of the new axiom to the knowledge base makes nothing inconsistent. Furthermore, in order to expand the knowledge base, the new state description axiom must be observed. This is specified by the *observeFluent* predicate which indicates that the information is new to the knowledge base and the knowledge base must be appropriately revised to accommodate the new information. Equation 2.18 shows the new state description axiom which will be true in the next state.

$$holds(isLoaded, s_1) \qquad (2.18)$$

Equation 2.19 shows the potential belief state after expansion. The whole process

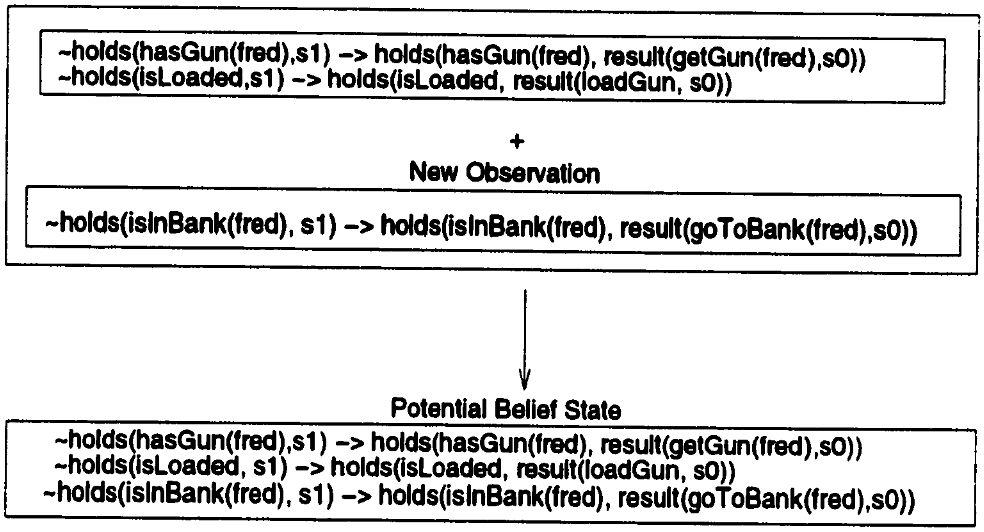


Figure 2.3: Expansion with an action effect axiom.

of expanding a knowledge base with a fluent is graphically shown in Figure 2.2.

$$holds(isInBank(fred), s_1) \tag{2.19}$$
$$holds(hasGun(barney), s_1)$$
$$holds(isLoaded, s_1)$$

Expansion is not limited to the incorporation of new state description axioms into our knowledge base. The belief state may be expanded by any axiom which we have defined to exist. Thus, action effect axioms, describing the pre and post conditions of actions may be added. In fact whole new plans may be added as well.

To illustrate further, Figure 2.3 shows how a knowledge base is expanded with a new action effect axiom. Such actions may be appended to the knowledge base at any time of the plan recognition reasoning process. The initial belief state contains just two action effect axioms

$$\neg holds(hasGun(fred), s_0) \rightarrow holds(hasGun(fred), result(getGun(fred), s_0))$$
$$\neg holds(isLoaded, s_0) \rightarrow holds(isLoaded, result(loadGun, s_0)) \tag{2.20}$$

If we expand the knowledge base with a new axiom, then this axiom will appear in the belief state which occurs after the expansion. We may use the *observedAction* predicate to append the new action to our knowledge base. Let

$$\neg holds(isInBank(fred), s_0) \rightarrow \\ holds(isInBank(fred), result(goToBank(fred), s_0)) \tag{2.21}$$

be the newly expanded axiom. The new potential belief state will contain the axioms from the initial belief state in Equation 2.20, as well as the new axiom:

$$\neg holds(hasGun(fred), s_0) \rightarrow \\ holds(hasGun(fred), result(getGun(fred), s_0))$$

$$\neg holds(isLoaded, s_0) \rightarrow holds(isLoaded, result(loadGun, s_0)) \tag{2.22}$$

$$\neg holds(isInBank(fred), s_0) \rightarrow \\ holds(isInBank(fred), result(goToBank(fred), s_0))$$

Figures 2.4 and 2.5 show how new plans are added to the knowledge base. Since plans are thought of as sequences of actions, the same plan may be defined in terms of different sequences of actions, if they all name the same goal. This is true with other axioms as well. For example action effect axioms describe preconditions which must hold true in order for some action to be executable. Each action is not limited however to a single set of preconditions and postconditions. It may happen that some action is permissible when some set of preconditions are fulfilled, resulting in certain postconditions. When some different set of preconditions is met, the same action may be permissible resulting in a different set of postconditions.

As is demonstrated by expansions on plans, the same plan may be added to the knowledge base (Figure 2.4) if it is uniquely different from plans already defined therein. Here the initial belief state consist of only one plan *toHunt* which is comprised of three actions.

$$plan(toHunt, [getGun(fred), loadGun, goToForest(fred)]) \tag{2.23}$$

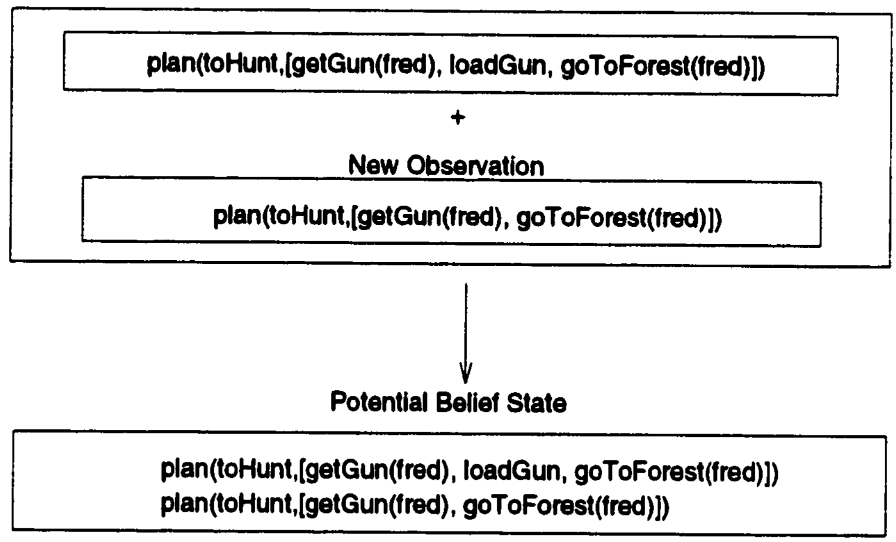


Figure 2.4: Expansion with a new definition of a plan.

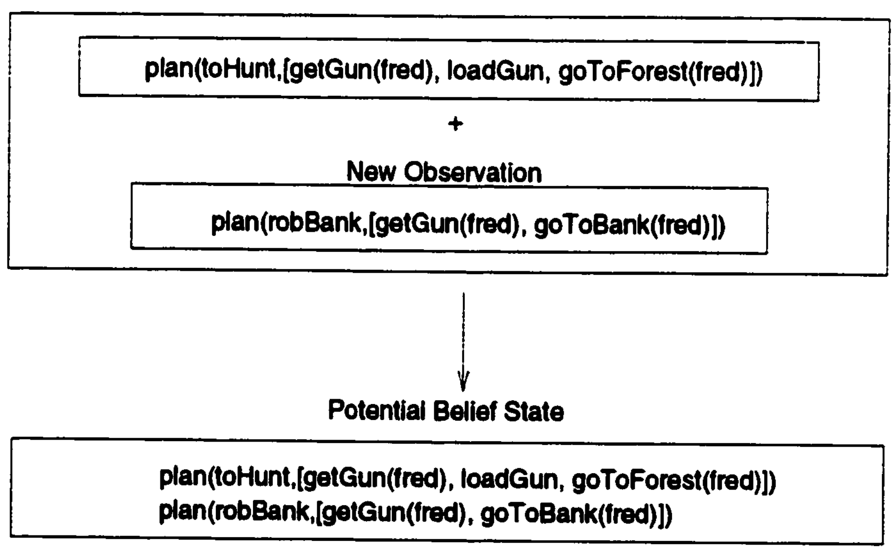


Figure 2.5: Expansion with a new plan.

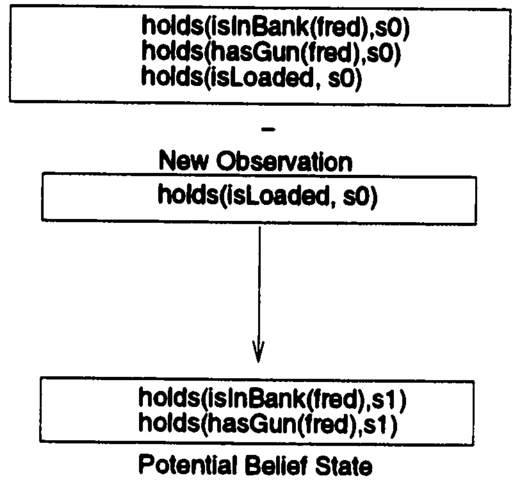


Figure 2.6: Contraction of a state description axiom.

This belief state may be expanded with a different plan *toHunt*, comprising of only two actions. We do not revise the original plan here and substitute it with the new plan, if we believe that both plans accurately subsume actions necessary to perform *hunting*. Therefore

$$plan(toHunt, [getGun(fred), goToForest(fred)]) \tag{2.24}$$

is a different version of the plan *toHunt* which is added to our original belief state. The resulting belief state accommodates the old and the new information, by listing both plans. This plan expansion can be viewed as revision without the existence of contradictory information.

$$\begin{aligned} &plan(toHunt, [getGun(fred), loadGun, goToForest(fred)]) \\ &plan(toHunt, [getGun(fred), goToForest(fred)]) \end{aligned} \tag{2.25}$$

In the same way, a completely new plan may be added to the knowledge base. This is demonstrated in Figure 2.5.

2.5.2 Contraction

As defined previously (Section 2.2.1), contraction deals with the removal of information from the knowledge base. The removed information is no longer believed to be

```
~holds(hasGun(fred),s1) -> holds(hasGun(fred), result(getGun(fred),s0))
~holds(isLoaded, s1) -> holds(isLoaded, result(loadGun, s0))

-

New Observation

~holds(hasGun(fred), s1) -> holds(hasGun(fred), result(getGun(fred), s0))

Potential Belief State

~holds(isLoaded, s1) -> holds(isLoaded, result(loadGun, s0))
```

Figure 2.7: Contraction of an action effect axiom.

true. Contraction may be therefore, in the strictest sense, thought of as the opposite operation of expansion. The original knowledge base contains some axioms, which after contraction no longer appear in the resulting knowledge base. We assume that any axiom which is representable may be contracted.

Figure 2.6 shows the simplest example of contraction. Here a state description axiom is removed from a belief state, because it no longer holds true. The initial belief state contains three state description axioms, and the final belief state contains only two of them. Information is therefore lost. This loss of information may be caused through the observation of the world or through observation of the agent performing some action which changes the description of the world.

Similarly, action effect axioms may be removed from the belief state. This type of removal decreases the number of such axioms in the resulting belief state. Whole plans may also be removed as shown in Figure 2.8.

2.5.3 Revision

The revision operator is quite different from the last two operators described. We assume that revision is used in cases where information is neither necessarily contracted from the belief set, nor is the belief set necessarily expanded. After the modification

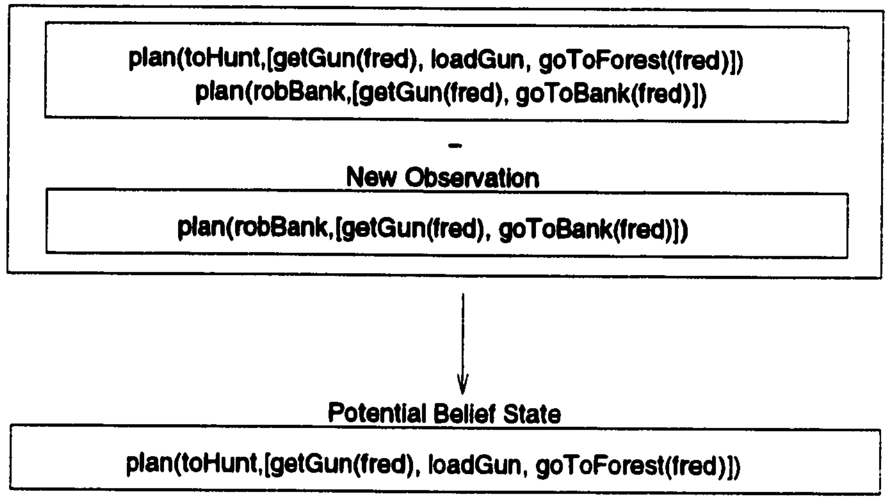


Figure 2.8: Contraction of a plan.

has taken place, there may result different possible mutually exclusive belief sets. The ambiguity of possible belief states, which results from the revision operation, is caused by the creation of choices of performing the revision. It may be impossible to determine which revision technique results in the *"most correct"* future belief state. Had this information been provided via some statistical measurement, utility function or epistemic entrenchment value [14], it might be possible to make a choice. In most cases such a choice is nevertheless impossible to make, and if made, would be paramount to guessing, with the consequence of loss of potentially important information.

Let us examine three examples to illustrate the peculiarities of the belief revision operation. Assuming an initial belief state K_0 to contain only three state description axioms

$$\begin{aligned} &holds(isInBank(fred), s_0) \qquad (2.26)\\ &holds(hasGun(fred), s_0)\\ &holds(isLoaded, s_0) \end{aligned}$$

we wish to revise it with the negation of the first axiom. We may not simply expand our belief state with this new axiom, as we have done previously, because the

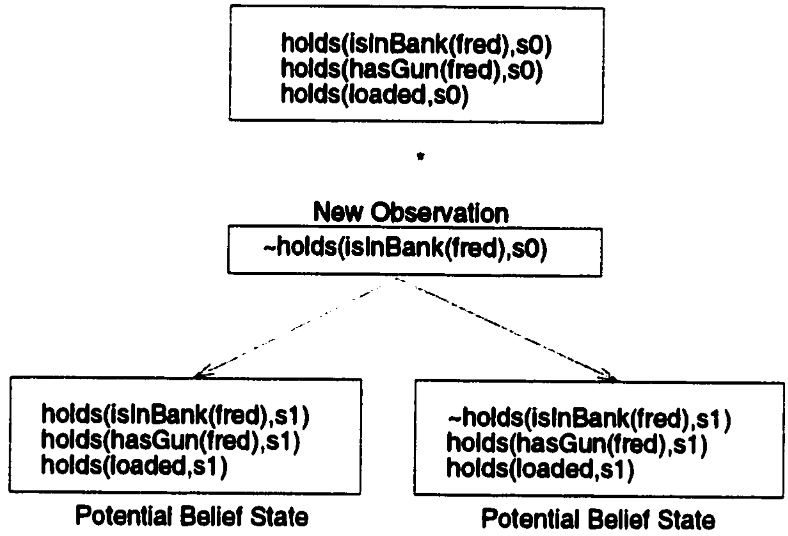


Figure 2.9: Revision of a state description axiom.

new axiom contradicts the previously believed axiom. This type of revision occurs each time we entertain a new piece of knowledge which contradicts one of the believed axioms and thus results in an inconsistency. If we observe

$$\neg holds(isInBank(fred), s_0) \tag{2.27}$$

we are forced to somehow revise our original belief state in such a manner as to eliminate the inconsistency which would otherwise result from adding the observation to our belief state. It turns out that there are at least two possible interpretations which may be used to perform this revision.

1. We can disregard the newly observed axiom and output a belief state identical to our original belief state. In this case, revision would be guided by our original axiom $holds(isInBank(fred), s_0)$ having a larger epistemic entrenchment value than the observed axiom $\neg holds(isInBank(fred), s_0)$.

2. We can contract the inconsistent axiom from our original belief state, and expand the state with the newly observed axiom. This revision operation assumes that the epistemic entrenchment of the new observation is higher than that of the original inconsistent axiom.

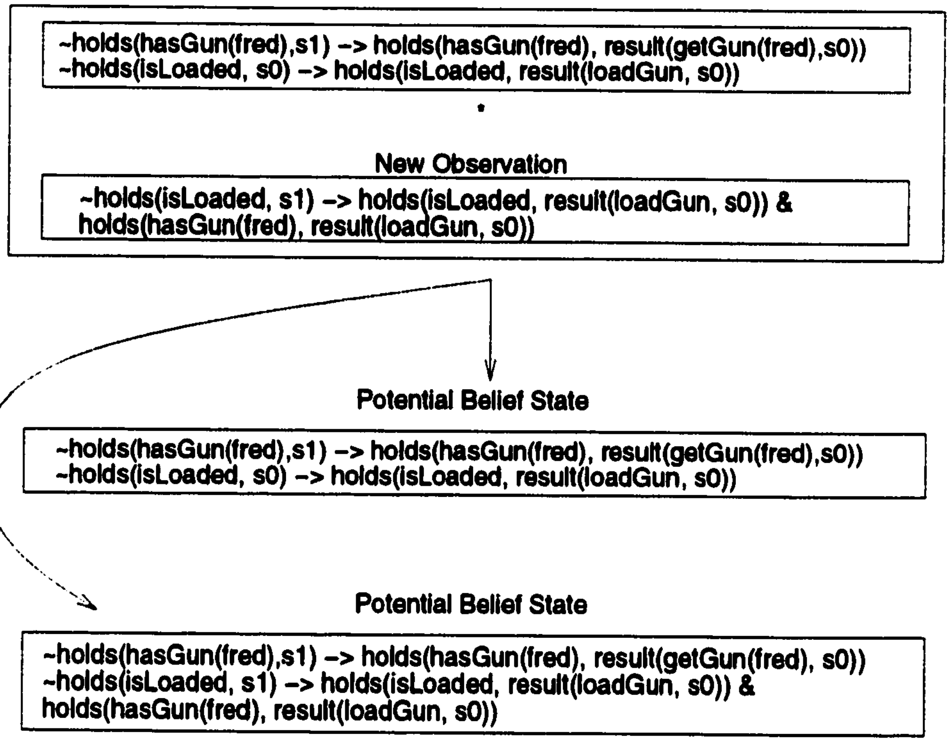


Figure 2.10: Revision of an action effect axiom.

In both cases the same amount of information is discarded. Which potential belief state do we then hold as true? In the AGM reasoning framework, we would have been forced to make a choice and use the result in subsequent knowledge maintenance operations. The Ghose-Goebel reasoning framework does not require us to do this. Instead, we maintain that it would be most frugal and correct to maintain both potential belief states, and only dispense of one of them when sufficient new information allows us to do this.

This approach is costly, but it has the benefit of minimal informational loss and the possibility to reach any potential future belief state. Figure 2.10 demonstrates the revision operator used on action effect axioms. It is noteworthy that here, as well as with plan revision in Figure 2.11, multiple possible potential belief states result. We may not discard any of them without loss of information.

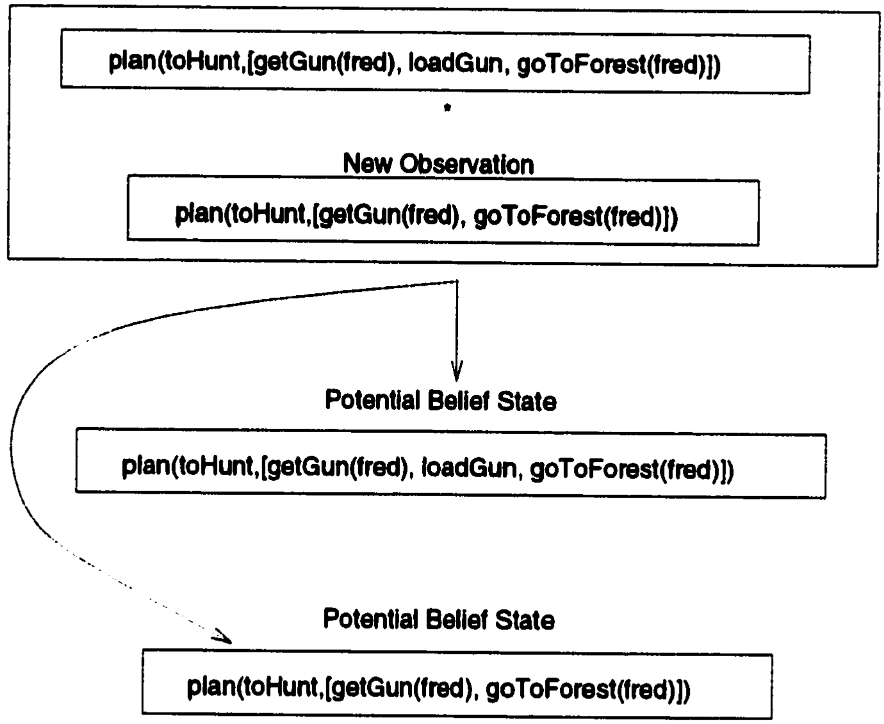


Figure 2.11: Revision of a plan.

2.6 Entrenchment of beliefs

The Ghose-Goebel belief change model maintains a dynamic set of assertions that represent actions and relationships amongst actions and plans. This method explicitly supports reasoning with incomplete knowledge and for revision of the belief base with newly observed facts. All information, however improbable, is maintained in some knowledge base, as long as there is a chance that it may become useful in some future state. The result of multiple revision operations, yields a large number of mutually inconsistent knowledge bases. Some of these are more probable of reflecting the true nature of the world than some others [5].

Some state description axioms have a higher probability of conforming to a realistic view of the world. Such fluents are said to have a higher *epistemic entrenchment* than other fluents. The value of the entrenchment may be used to rank axioms within each knowledge base and to rank knowledge bases, which result from belief revision operations.

There is no single way to determine the entrenchment value of individual beliefs, because such calculations are based on context of each knowledge base. Furthermore, entrenchment calculations may be made based on various criteria, including:

Probability: The probability of a fluent being true. Let f_i and f_j be two fluents which hold in state S. If $P(f_i, S) \geq P(f_j, S)$, then the axiom denoted by f_i is said to be at least as epistemically entrenched as the axiom denoted by f_j in state S.

Utility: The utility of a fluent. This might be a very unrealistic criterium for determining entrenchment. By utility, we mean the usefulness of the fluent. The utility might very well differ from probability and something quite useful, might be very improbable. Let $U(f_i)$ denote a measure of utility of fluent f_i in state S. If $U(f_i, S) \geq U(f_j, S)$, then the axiom denoted by f_i is said to be at least as epistemically entrenched as the axiom denoted by f_j in state S.

Reliability: The reliability (credibility) of the source. This criterium only considers the source of information. The motivation lies in the fact that we are more likely to believe a fact learned from a reliable source, than one learned from a less reliable source. Let $R(f_i)$ denote a measure of reliability of the source of fluent f_i in state S. If $R(f_i, S) \geq R(f_j, S)$, then the axiom denoted by f_i is said to be at least as epistemically entrenched as the axiom denoted by f_j in state S.

Peter Gärdenfors in [5] specifies postulates for epistemic entrenchment. Let $f_i > f_j$ represent the notion that f_i is more epistemically entrenched than f_j. The five postulates are presented in equations 2.28 to 2.32.

$$If \;\; f_i \leq f_j \;\; and \;\; f_j \leq f_k, \;\; then \;\; f_i \leq f_k \tag{2.28}$$

The justification of this *transitivity* postulate (2.28) is motivated by the assumption that entrenchment is a function of linear measurement. If an axiom f_k is at least as entrenched as some other axiom f_j, then f_k is also at least as entrenched as a third axiom f_i, of lower or equal entrenchment to f_j.

$$If \;\; f_i \vdash f_j, \;\; then \;\; f_i \leq f_j \tag{2.29}$$

The rationality of this *dominance* postulate (2.29) draws from the contraction operation. If either f_i or f_j must be contracted from K, and f_i entails f_j, then f_i ought to be contracted. This follows from the reasoning that if we contract f_j, it is still entailed in f_i, and thus f_i must also be contracted in addition to f_j. This reasoning is counter intuitive, but required for the presentation of the other three postulates [5]. The counter intuitiveness derives from preserving the postulate of informational economy which states that minimal amount of information should be discarded when performing belief change operations. Since the contraction of f_i discards more information than the contraction of f_j, informational economy does not seem to be preserved in this case.

$$For \; any \;\; f_i \;\; and \;\; f_j, \;\; f_i \leq (f_i \wedge f_j), \;\; or \;\; f_j \leq (f_i \wedge f_j) \tag{2.30}$$

This *conjunctiveness* postulate (2.30) follows directly from postulate 2.29.

$$When \;\; K \neq K_{\perp}, \;\; f_i \notin K \;\;\; iff \;\;\; f_i \leq f_j, \;\; for \; all \;\; f_j \tag{2.31}$$

This *minimality* postulate (2.31) requires that axioms not in the knowledge base have a lower epistemic entrenchment than the axioms in the knowledge base.

$$If \;\; f_j \leq f_i \;\; for \; all \;\; f_j, \;\; then \;\; \vdash f_i \tag{2.32}$$

The final *maximality* postulate (2.32) follows from postulate 2.29, since if $\vdash f_i$, then $\forall f_j \mid f_j \vdash f_i$.

Chapter 3

The plan recognition system

3.1 Purpose

To help make our intuition clear, and as a basis for elaborating specific details, we begin by assuming that any plan recognition system has to be able to *both* accept observations about a changing world, and make predictions about what could plausibly take place in that world. To help make these ideas concrete, we first assume that we could use a goal-directed logic programming reasoning system like Prolog to define the top level functionality of a generic belief revision system.

This generic system would have to interpret the following relations:

- *observe(Action, Situation)* We can assert that a particular action has been observed in a particular situation.
- *predict(Action, Situation)* We can query the system to determine if it is reasonable to expect a certain action in a particular situation.

Note that the idea of being able to predict the next anticipated action can be elaborated to the more familiar plan recognition relation as follows: if we assume that plan recognition takes place in the context of a plan library consisting of named sequences of actions, e.g.,

$$plan(planName, [a_1, a_2, ..., a_n]) \tag{3.1}$$

then we can generalize $predict(Action, Situation)$ to the more familiar output of conventional plan recognition systems, namely

$$predictPlan(Plan, Situation) \tag{3.2}$$

We expect our non-monotonic reasoning system to assume consistent hypotheses based on observations, and to provide us with the names of plans that could plausibly be considered as those intended by an agent that had carried out some number of the actions in the given situation.

Here we want the underlying belief revision system to process observations as expansions, contractions, or revisions, and then be able to produce predictions based on existing plan libraries. The predictions could be in the form of guessing what action might be next attempted by an observable agent, or in the form of a named plan, whose actions somehow consistently subsume those already observed.

To be a little more specific, we can begin with the following definitions:

- $observedAction(Action, Situation)$. True when Action has been observed in Situation, e.g., $observedAction(getGun(fred), s_1)$.

- $observedFluent(Fluent, Situation)$. True when Fluent has been observed in Situation, e.g., $observedFluent(isInBank(fred), s_3)$.

- $predictAction(Action, Situation)$. True if there is a belief state consistent with Situation and Action is possible in that state. e.g., $Action = goToBank$, $Situation = result(getGun(fred), s_2)$

- $predictPlan(Plan, Situation)$. True if the plan library contains a definition of $Plan$ in a form which associates Plan with a list of actions which comprise it: $plan(Plan, [a_0, a_1, \ldots, a_n])$. Furthermore, the preconditions or consequences of actions comprising the plan but not yet observed must hold or be assumable.

To explore the ideas of how non-monotonic reasoning plays a role in plan recognition, note that if the preconditions or consequences of unobserved actions are at least assumable, the the plans associated with those actions are plausible.

$$predictPlan(Plan, Situation) \Leftarrow \tag{3.3}$$
$$(plan(Plan, [a_0, a_1, \ldots, a_i, a_{i+1}, \ldots, a_n])),$$
$$observedOrAssumedAll([a_0, a_1, \ldots, a_i, a_{i+1}, \ldots, a_n],$$
$$Situation)$$

If the *actions* $a_1, \ldots, a_i$ have been observed, it may be non-monotonically feasible to assume that *Plan* is a plan if it can be verified that the actions $a_{i+1}, \ldots, a_n$ may be taken from *Situation*, or that the consequences of required but unobserved actions are themselves assumable. As long as no information contradicts such assumptions for actions $a_{i+1}, \ldots, a_n$, *Plan* is plausible. This is analogous to saying that the enabling conditions and affects are not required to hold or be observed, but only to be possible. This is true if nothing would make these actions impossible [13].

3.2 Various approaches for plan recognition

3.2.1 Recognizing Kautz' plans

Henry Kautz provides a theory of plan recognition capable of handling concurrent actions, shared steps between actions and disjunctive information. His theory views plan recognition in terms of deductive inference based on a set of observations, a plan library, and a set of constraints. He presents two underlying assumptions:

1. The known ways of performing an action are the only ways, meaning that all ways of performing it are known;

2. All actions are purposeful and all possible reasons for performing an action are known.

Kautz' framework introduces an action taxonomy which serves as an exhaustive description of ways in which actions can be performed. This taxonomy specifies that

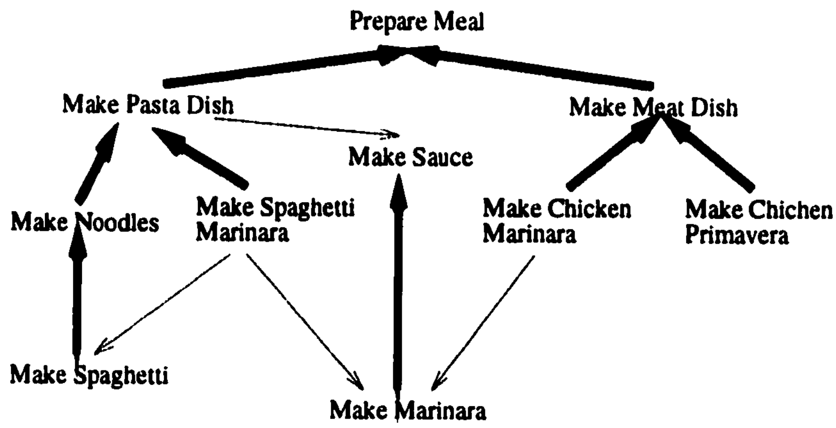


Figure 3.1: A Subset of Kautz' Action Hierarchy

an action can be used as a step of a more complete action. The action representation is based on two hierarchies:

1. *A specialization hierarchy* (abstraction hierarchy) which subdivides action types top to bottom with more specific actions being instances of more general actions. For example in figure 3.1 *an instance of making a pasta dish* is also an instance of *preparing a meal.*

2. *A decomposition hierarchy* indicates the necessary conditions for an action instance to occur. For example in figure 3.1 an instance of *making spaghetti marinara* consists of at least *making spaghetti* and *making marinara.*

The plan library is encoded with axioms. Both of Kautz' hierarchies are encoded. An example of a specialization hierarchy axiom is encoded as:

$$\forall e \;\; \varphi_1(e) \supset \varphi_2(e) \tag{3.4}$$

Here φ_1 and φ_2 are plans (action types) such that φ_1 is a specialization of φ_2. In figure 3.1 φ_1 could be represented by the action type *Make Pasta Dish*, while φ_2 by *Prepare Meal.*

The decomposition hierarchy divides each plan φ into its components $\sigma_1 \ldots \sigma_n$. These have associated with them action types $\psi_1 \ldots \psi_n$, since each plan component is viewed as an action. An axiomatized decomposition hierarchy consists of axioms of the form:

$$\forall e \;\; \varphi(e) \supset \psi_1(\sigma_1(e)) \wedge \ldots \wedge \psi_n(\sigma_n(e)) \tag{3.5}$$

In reference to figure 3.1 we decompose the plan *Make Spaghetti Marinara* as consisting of two actions *Make Spaghetti* and *Make Marinara*. Assume that an *o occurs* predicate is introduced. Formally, this decomposition could be axiomatized as:

$$\begin{aligned} & o(Plan, MakeSpaghettiMarinara) \supset \\ & o(Action, MakeSpaghetti) \wedge o(Action, MakeMarinara) \end{aligned} \tag{3.6}$$

Since Kautz' assumptions deal with a closed world scenario, plan recognition is accomplished by finding a sequence of actions which in a *thrift* (smallest cardinality from the leaf of the plan library to the recognized plan within the tree structure of which the plan library comprises) sense describe the plan. According to Kautz' assumptions, no specializations are present within the plan library which are not specified and non-leaf plans (action types which can be decomposed) are in fact decomposable into simpler plans. These assumptions are achieved by Kautz by performing a sequence of nonmonotonic circumscriptive minimizations of axioms which maintains the closed world assumptions [9, 21].

Kautz views plan recognition as deductive inference which is based on an action taxonomy (specialization and decomposition hierarchies). His approach is different from ours because of two very significant assumptions. In his framework:

1. All known ways of performing an action are known and recorded in his action hierarchy. This implies that the stated actions are the only ones which constitute plans. Our method of plan recognition is based on a dynamically changing belief maintenance system, where only a small subset of the plan library is known and

as new information becomes available, it is incorporated into the knowledge base.

2. All actions are purposeful and all possible reasons for performing an action are known. Our framework allows for actions which are irrelevant (are observed, but are not part of any known plan) and redundant (the same action may occur more than once without an effect on the resulting state(s)).

Thus the key difference of Kautz's approach lies in his plan library being static. The advantages of this approach lie with a well defined search space and a closed world scenario. The disadvantages lie in the approach not reflecting the true nature of the changing environment which our PR architecture accounts for using BR operators and GG belief maintenance.

3.2.2 Vilain's plan parsing

Marc Vilain [21] uses the Kautz' plan recognition model to suggest that plan recognition is in reality similar to parsing text. He provides a relationship of Kautz' model to existing grammatical frameworks and recognizes areas where parsing may be efficiently used to perform plan recognition.

Vilain argues that since parsing is a well understood problem which possesses efficient solutions, it follows than that when applying parsing techniques to plan recognition, similar efficient techniques can be found. His approach to plan recognition by parsing involves:

- The construction of a *context free grammar* to represent plan libraries. It is important to note that these grammars allow acyclic hierarchies, which Kautz' plan library does not. This is significant because recursive plan definitions may be represented. Furthermore, actions which are not necessarily *strongest* in the context of Kautz may be utilized to recognize a plan.

- The application of an extension of Earley's context-free recognition algorithm to perform parsing of the grammar to locate plans. The parse trees obtained by the algorithm could be interpreted as first-order logic statements.

Briefly, Earley's algorithm works by analyzing all observations and adding grammatical rules based on derived constituents of the hierarchies. It generates intermediate parse trees by certain rule manipulations and modifies the grammar. It consists of three phases: prediction, scanning and completion. Upon completion a set of rules has been created which is capable of recognizing plans.

Marc Vilain's [21] approach stipulates the following propositions:

1. Under the sentential interpretation of parse trees, Earley's algorithm computes the minimal covering models of a base-level observation with respect to a decomposition hierarchy with ordered unshared steps [12].

2. There is a $O(n^3)$ time plan recognition algorithm for hierarchies with ordered, unshared steps and for disjunctive or abstract observations.

3. Recognizing plans with abstraction and partial step order is NP-Complete, regardless of the recognition tactic.

This approach shows the aspects of Kautz' [9] plan recognition methods which can be performed by parsing and which achieve gains in tractability [21].

3.2.3 Charniak's probabilities model

Charniak and Goldman [3] argue that plan recognition is a problem dealing with inference under uncertainty. They show a method of solving the plan recognition problem which is based on Bayesian probability theory. Their solution consists of:

1. Retrieving candidate plan explanations.

2. Assembling the explanations into a Bayesian plan recognition network. The network represents a probability distribution over the possible explanations.

3. Updating of the network to make a selection of the *strongest* plan explanation, based on the *most likely* interpretation.

The authors utilize two distinct knowledge representation methods. One is used to deal with plans and actions and is based on predicate calculus. The other deals with representing probabilities in terms of the Bayesian networks. Such a network contains nodes and edges between them. Probabilities are assigned to the edges and these represent various interactions between actions. The nodes represent various random variables. The authors use the diagnosis example to demonstrate the differences between nodes and edges. Various diseases would be represented by the nodes, and the edges would direct these towards symptoms which might be caused by the diseases.

The network is created by using clauses. Objects may be introduced into a network if proper evidence exists. Five main clauses are used to build the network and traverse through it. Once a network has been created, rules are used to perform plan recognition. These include proposing a plan hypothesis, creating a plan hypothesis and linking actions with a plan.

The authors treat actions and plans equally. Plans, in their view, are simply more complex actions which consist of other actions. Plan recognition is achieved by evaluating the Bayesian networks and selecting the most likely plan.

The authors claim at least three advantages of the probabilistic approach to plan recognition over Kautz' method [3, 9]:

1. It is claimed that since Kautz' methods are based on the concept of finding minimal set covers, if two or more plan explanations exist, Kautz' technique is incapable of selecting the *most likely* explanation. In our opinion this is not necessarily a drawback. A *most likely explanation* may not be *the* desired explanation.

2. Finally, abduction and set minimization are said to be incompatible. This is a very strong statement, which definitely holds in the area of diagnosis. Sometimes it may be required to stipulate all possible diseases instead of just one,

based on the symptoms. We agree with such an approach because our method also provides all of the possible methods, as long as they are possible, but not necessarily most likely.

The key to the probabilistic approach lies in the rules which translate plan recognition problems into Bayesian networks.

3.2.4 Lin's abductive model

Lin discusses plan recognition as an application of obvious abduction. He utilizes a plan hierarchy network to represent domain knowledge about plans which is based on Kautz' network of Figure 3.1. Instead of having only specialization and decomposition links, as does Kautz, he introduces a third type of relationship, the *isa* link. These represent special cases of the specialization hierarchy. In Figure 3.1, there exists a specialization link between "make spaghetti" and "make noodles" because making spaghetti is a special case of making noodles. There is an inverse *isa* link since making noodles is a form of making spaghetti.

The inputs to the plan recognition system are comprised of observations about the actions an agent has taken. They are represented as leaf nodes in the hierarchy. The best examples from our diagram are "make spaghetti" and "make marinara". Plan recognition in Lin's framework selects a subset of the plan library which contains the observed actions, together with some higher level goal. The goal is represented as a higher level node in the hierarchy, an example of which would be "prepare meal" in Figure 3.1. The links from the actions to the goal provide a scenario of actions which are recognized to satisfy the goal. The paths such generated represent the recognized plans and their action constituents [12].

Lin's obvious abduction method makes assumptions regarding the relationship amongst possible actions, so that the plan recognition algorithm is a computation. His method selects the most probable plan, under an interpretation similar to Charniak's and Goldman's.

Chapter 4

Plan recognition competence

4.1 Overview

In order to fully understand our merger of the techniques underlying plan recognition, belief revision and non-monotonic reasoning, it is important to note several assumptions.

Plan recognition competence is achieved by separating our knowledge maintenance system from the plan recognition system. The knowledge maintenance system is responsible for propagation of belief change operations, maintenance of possible potential belief states and non-monotonic assumptions as to the possibility of an action occurring with no evidence to the contrary. The plan recognition system is based on a goal-directed logic programming system, where the information provided by the knowledge base is used to define its top-level functionality.

4.2 Expected behaviour

The plan recognition system has to interpret the following relations:

1. $assumable(Action, Situation)$ We can assert that a particular action may be executed in some future situation. This behaviour is achieved by checking the preconditions of the action. If there do not exist any preconditions which are

in conflict with the state description axioms in the knowledge base, then the specified action may be executed in some future situation.

This is a very powerful relation since it does not require all of the action's preconditions to be fulfilled. The only criterium which is taken into consideration is that no contradictory information is present in the knowledge base describing the specified situation [9]. We are not concerned whether an action can be executed now; we non-monotonically assume those of the action's preconditions which are not currently supported by the knowledge base, as long as, they are not contradicted by information we already have. This allows us to treat the action as possibly executable in some future state.

The drawbacks of this assumption lie in the possibility of the assumed preconditions becoming false in the future. This can happen if the preconditions become contradicted by newly acquired information.

For example, let $Cn(K_S)$ be the set of the consequences of the axioms of the knowledge base in situation S, then

$$assumable(a_i, S) \tag{4.1}$$

is *True* if and only if

$$Cn(K_S) \cup preconditions(a_i) \nvdash False \tag{4.2}$$

2. $assumable([a_1, a_2, \ldots, a_n], Situation)$ We can assert that the specified sequence of actions may be executed from some future situation. The difference between this and the previous relation lies in the possibility of determining whether a sequence of actions could possibly be executed. This behaviour is achieved by checking the preconditions of each action. If there do not exist any preconditions which are in conflict with the state description axioms in the knowledge base, then the specified action may be executed in some future situation. This is repeated for each subsequent action specified in the relation.

As above, this is also a very powerful relation because it does not require all of the actions' preconditions to be fulfilled. The only criterium which is taken into consideration is that no contradictory information is present in the knowledge base describing the specified situation. We are not concerned whether an action can be executed now; we non-monotonically assume those of the actions' preconditions which are not currently supported by the knowledge base, as long as, they are not contradicted by information we already have. This allows us to treat the action as possibly executable in some future state. Again the drawbacks of this assumption lie in the possibility of the assumed preconditions becoming false in the future. This can happen if the preconditions, become contradicted by newly acquired information. Note that this assumes a chain or sequence of assumptions regarding preconditions.

For example, let $Cn(K_S)$ be the set of the consequences of the axioms in the knowledge base in situation S, then

$$assumable([a_1, a_2, \ldots, a_n], S) \tag{4.3}$$

is *True* if and only if

$$Cn(K_S) \cup preconditions([a_1, a_2, \ldots, a_n]) \not\vdash False \tag{4.4}$$

3. *possible(Action, Situation)* We can assert whether an action may be executed in the specific situation. If the preconditions of *Action* are fulfilled in *Situation*, then the relation is *True*, otherwise *False*. A way to increase the usefulness of this relation, is for the relation be defined over the list of preconditions which have failed for *Action*. This addition could serve as a specification for actions which have to still take place in order for *Action* to be executable in the specified situation. When the returned list of unfulfilled preconditions would be satisfied, then and only then could *Action* be executed.

For example, let $Cn(K_S)$ be the set of the consequences of the axioms in the knowledge base in situation S, then

$$possible(a_i, S) \tag{4.5}$$

is *True* if and only if

$$preconditions(a_i) \in Cn(K_S) \tag{4.6}$$

4. $possible([a_1, a_2, \ldots, a_n], Situation)$ We can assert whether a sequence of actions may be executed from the specified situation. If all preconditions of $[a_1, a_2, \ldots, a_n]$ are fulfilled in $Situation$, then the relation would return a boolean $True$, otherwise a $False$ would be returned. To be more specific, the preconditions of each action specified in the list $[a_1, a_2, \ldots, a_n]$ of actions has to be fulfilled in the knowledge base.

   The definition of this relation could be expanded for the relation to return the list of preconditions which have failed for all actions in $[a_1, a_2, \ldots, a_n]$. This addition could serve as a specification for actions which have to still take place in order for each *Action* in the list to be executable in the specified situation. When the returned list of unfulfilled preconditions would be satisfied, then and only then could the plan $[a_1, a_2, \ldots, a_n]$ be executed from $Situation$.

   For example, let $Cn(K_S)$ be the set of the consequences of the axioms in the knowledge base in situation S, then

$$possible([a_1, a_2, \ldots, a_n], S) \tag{4.7}$$

   is *True* if and only if

$$preconditions([a_1, a_2, \ldots, a_n]) \in Cn(K_S) \tag{4.8}$$

5. *predictable(Action, Situation)* We can predict *Action* in the specified situation. Perhaps the simplest form of prediction takes place when *Action* is simply possible in the specified situation. Here, we mean 'simplest' because we predict every action which is possible. Thus one interpretation is:

$$predictable(a_i, S) \Leftarrow possible(a_i, S) \tag{4.9}$$

Yet another interpretation takes into account the non-monotonic nature of plan recognition. Predictability may be defined on those actions which we assume may occur in the future. This interpretation opens up the possibility of predicting actions based on their preconditions, as long as these are not contradicted by any knowledge in the knowledge base. Thus

$$predictable(a_i, S) \Leftarrow assumable(a_i, S) \tag{4.10}$$

We could also combine the previous two definitions to yield a third interpretation, which perhaps serves as the most comprehensive. If we strive to provide a competent theory of plan recognition, the aim should be to predict any feasible future action. This implies actions which are possible now or are assumable in some future state. The interpretation of predictability could therefore be stated as:

$$predictable(a_i, S) \Leftarrow possible(a_i, S) \vee assumable(a_i, S) \tag{4.11}$$

See the examples in section 4.4.

One way of implementing this relation, could involve the usage of the action-effect axioms which are included in the current knowledge base to retrieve a set of preconditions for all actions defined for the specified situation. Then for each action defined in the action-effect axioms, a match could be instituted against the state description axioms to find which preconditions are fulfilled. If all of the preconditions for the actions were fulfilled, than that action is said to be executable from the current situation.

6. *predictable(Plan, Situation)* We can predict whether the specified plan may be executed in the specified situation. This relation may be implemented in

terms of a set of $predictable(a_i, S)$ relations where each $a_i \in Plan$. If $Plan = [a_1, a_2, \ldots, a_n]$, then $predictable(Plan, Situation)$ may be determined using any of the interpretations for predictability as discussed in examples 4.9, 4.10, and 4.11. Therefore:

$$\begin{aligned}&predictable([a_1, a_2, \ldots, a_n], S) \Leftarrow\\&possible([a_1, a_2, \ldots, a_n], S) \vee\\&assumable([a_1, a_2, \ldots, a_n], S)\end{aligned} \tag{4.12}$$

predicts all plans which are possible or assumable in state S.

The significance of this relation lies in the fact that it actually determines whether $Plan$ is possible by non-monotonic assumption of the preconditions of actions which constitute the $Plan$.

7. *predictPlans(Plans, Situation)* We can predict a set of plans which are possible in the specified situation. This is an extension of the previous relation to retrieve all possible plans, not just the specified one. It could be implemented by means of multiple $predictable([a_1, a_2, \ldots, a_n], S)$ relations, each corresponding to a different plan defined in the knowledge base. If $Plans = \{P_1, P_2, \ldots, P_n\}$, then

$$\begin{aligned}&predictable(Plans, S) \Leftarrow\\&predictable(P_1, S) \wedge\\&predictable(P_2, S) \wedge\\&predictable(\ldots, S) \wedge\\&predictable(P_n, S)\end{aligned} \tag{4.13}$$

Each plan P_i is just a sequence of actions which constitute it, where a_j^i denotes the j^{th} action in the i^{th} plan.

$$predictable(P_i, S) \Leftarrow predictable([a_1^i, a_2^i, \ldots, a_n^i], S) \tag{4.14}$$

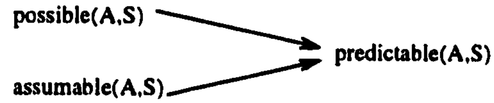


Figure 4.1: The plan recognition relational hierarchy.

4.3 Plan recognition relational hierarchy

Before going on with a demonstration of how each of these relations could be employed in pursuit of plan recognition and action prediction, it is important to note the hierarchy which exists among the relations. This hierarchy may serve as a guideline as to the generality and specificity of information which could be gathered from the knowledge base. It distinguishes possible actions from assumable ones.

The two relations on the left of Figure 4.1 are used in the definition of the relation on the right. The $possible(A, S)$ and $assumable(A, S)$ relations return truth values based on actions' preconditions. The $predictable(A, S)$ relation makes use of the truth values returned, to establish the feasibility of the specified action or plan. Note that the diagram simply presents the simplest case of a single action in a state. This could be analogously extended to plans.

4.4 Demonstration of possibilities

In order to demonstrate the plan recognition aspects of our reasoning, we must come up with a feasible state of the world described by state description axioms, action-effect axioms and plan definitions. Assuming that all required frame axioms and state axioms are also defined, which allow us to reason in a goal-oriented manner, let our initial state of the world consist of the following state description axioms: $S_0 = \{f_1, f_2, f_3\}$, where f_i are fluents such as $holds(isInBank(fred), s_0)$. In addition, the following table lists the action-effect axioms subdivided into their three constituent parts.

Preconditions	Action	Postconditions
f_1, f_2	a_1	$\neg f_1, f_2$
$\neg f_1, f_3$	a_2	$\neg f_1, f_4$
f_3, f_4	a_3	$\neg f_1$
$\neg f_1$	a_4	$\neg f_1$

In order for the action a_1 to be executable, the fluents f_1 and f_2 must hold. After the action has executed the fluents $\neg f_1, f_2$ hold. The other actions behave similarly. In addition, the following plans are also defined in S_0.

Plan	Actions
P_1	a_1, a_2, a_3
P_2	a_3, a_2
P_3	a_4, a_2
P_4	a_1

Now, starting with the most specific relation $possible(a_1, S_0)$, we determine whether the specified action is possible. The following table demonstrates the behaviours of each relation. The predictability relation has been evaluated according to the definition specified in example 4.11.

Axiom	Result
$possible(a_1, S_0)$	$True$
$possible(a_2, S_0)$	$False$
$possible(a_3, S_0)$	$False$
$possible(a_4, S_0)$	$False$
$assumable(a_1, S_0)$	$True$
$assumable(a_2, S_0)$	$False$
$assumable(a_3, S_0)$	$True$
$assumable(a_4, S_0)$	$False$
$predictable(a_1, S_0)$	$True$
$predictable(a_2, S_0)$	$False$
$predictable(a_3, S_0)$	$True$
$predictable(a_4, S_0)$	$False$

The action a_1 is executable in S_0 because all of its preconditions are satisfied, without having to resort to non-monotonic assumptions about their possible future state. $preconditions(a_2) = \{\neg f_1, f_3\}$ and $S_0 = \{f_1, f_2, f_3\}$. Clearly $\neg f_1$ and f_1 are inconsistent and therefore the action a_2 may not be assumed to be executable in some future state, nor can it be executed in S_0. This is not the case with action a_3. The axiom f_4 does not hold in S_0, therefore the actions may not be currently executed. However, none of the currently held axioms f_1, f_2 nor f_3 is in conflict with f_4. The axiom f_4 may be assumed to possibly hold in some future state, hence the $assumable(a_3, S_0)$ returns a positive value. The action a_4 is neither assumable nor possible by the same reasoning as action a_2. It has been included in the table to demonstrate the applicability of entrenchment in plan selection which will be discussed in section 4.5.

The plural versions of these relations behave in a very similar way. Any plan containing a sequence of actions among which are a_2 and a_4 will not be predicted. This is due to the definition of predictability in example 4.11. The following table demonstrates the behaviours of each relation as pertaining to state S_0 and plans P_1,

P_2, P_3 and P_4. Note that actions are treated on an individual basis as if they were possible or assumable in state S_0.

Axiom	Result
$possible(P_1, S_0)$	$False$
$possible(P_2, S_0)$	$False$
$possible(P_3, S_0)$	$False$
$possible(P_4, S_0)$	$True$
$assumable(P_1, S_0)$	$False$
$assumable(P_2, S_0)$	$False$
$assumable(P_3, S_0)$	$True$
$assumable(P_4, S_0)$	$True$
$predictable(P_1, S_0)$	$False$
$predictable(P_2, S_0)$	$False$
$predictable(P_3, S_0)$	$False$
$predictable(P_4, S_0)$	$True$

If we treat the actions as aggregate sequences where the specified order is important, we end up with different possibility, assumability and predictability results. This is true, because after performing the first action in the sequence, the resulting state S_1 may be different from S_0, thus where the second action might have been feasable in state S_0, it may no longer be in state S_1. One interpretation of the aggregate view of sequences of actions may be:

Axiom	Result
$possible(P_1, S_0)$	$True$
$possible(P_2, S_0)$	$False$
$possible(P_3, S_0)$	$False$
$possible(P_4, S_0)$	$True$
$assumable(P_1, S_0)$	$True$
$assumable(P_2, S_0)$	$True$
$assumable(P_3, S_0)$	$False$
$assumable(P_4, S_0)$	$True$
$predictable(P_1, S_0)$	$True$
$predictable(P_2, S_0)$	$True$
$predictable(P_3, S_0)$	$False$
$predictable(P_4, S_0)$	$True$

Let us explain these results by examining plan P_1. This plan consists of three actions a_1 followed by a_2 and a_3. By executing action a_1 in state $S_0 = \{f_1, f_2, f_3\}$, one of the resulting states due to revision will be $S_1 = \{\neg f_1, f_2, f_3\}$. Clearly action a_2 is possible in this state since all of its preconditions are satisfied. It is also assumable because S_1 does not contain axioms which would contradict any preconditions of this action. The resulting state $S_2 = \{\neg f_1, f_2, f_3, f_4\}$ satisfies the preconditions of action a_3.

4.5 How can we predict better?

After careful examination of the definitions of assumability and possibility, it is possible to note that an action may not be possible and not assumable. This is true because the definition of possible asserts that all of the action's preconditions must hold in the current state. If they hold, then clearly they are not in contradiction with any of the preconditions, thus they must be assumable.

$$possible(a_i, S) \Rightarrow assumable(a_i, S) \tag{4.15}$$

Equation 4.15 is easily argued by examining the preconditions of a_i. If, however, an action is not possible in some state, it cannot be determined whether the action is or is not assumable.

We could tend to prefer predicting based on:

- The highest number of possible and assumable relations satisfied for each predictable action or plan.

  This criteria flows from satisfying the abstract notion of the informational economy postulate. It does not take any kind of entrenchment measurements into consideration and is simply based on the amount of information used to determine predictability. For example, the predictable relation 4.16 uses four fluents A_1 through to A_4 to predict the action A in state S. The predictable relation 4.17 of action B in state S, uses only three fluents B_1, B_2 and B_3. Given that a higher number of relations is used to predict the action A, this heuristic would prefer to predict A over B.

$$\begin{aligned} &predictable(A, S) \Leftarrow \\ &possible(A_1, S) \wedge \\ &possible(A_2, S) \wedge \\ &assumable(A_3, S) \wedge \\ &assumable(A_4, S) \end{aligned} \tag{4.16}$$

$$\begin{aligned} &predictable(B, S) \Leftarrow \\ &possible(B_1, S) \wedge \\ &possible(B_2, S) \wedge \\ &assumable(B_3, S) \wedge \end{aligned} \tag{4.17}$$

- The largest number of possible relations which are satisfied for each predictable relation, given at least two predictable relations with the same cardinality of comprising relations.

$$
\begin{aligned}
&predictable(C, S) \Leftarrow \\
&possible(C_1, S) \wedge \\
&possible(C_2, S) \wedge \\
&possible(C_3, S) \wedge \\
&assumable(C_4, S)
\end{aligned} \tag{4.18}
$$

Consider a belief state with two predictable relations 4.16 and 4.18. Note that the cardinality of both of these is 4 since each is comprised of a total of four possible or assumable relations. It is the case that in the predictable relation 4.16, two *possible* relations are used. In the case of the predictable relation 4.18, three *possible* relations are employed. Given, that in the latter case, a higher number of *possible* relations is used, we would tend to prefer to predict action C over action A.

The motivation here lies in the fact that in state S, only C_4 needs to be satisfied before the action can take place, versus two fluents A_3 and A_4 having to be satisfied.

- The belief state with the largest number of predictable relations.

The motivation for this heuristic lies in the fact that more information is used when we can predict a larger number of actions and plans. Note that due to revision, we may end up with several mutually exclusive belief states. Without considering any form of entrenchment, we may wish to prefer to predict based on the highest number of probabilities which are "makable." This will depend on each belief state.

4.5.1 Predicting based on entrenchment

So far, prediction has been limited to whether an axiom has been possible or assumable. This simple view need not be the only one which can be taken to recognize actions or plans. In section 2.6 the concept of epistemic entrenchment has been introduced, which strives to order axioms within a belief state according to utility, probability or reliability. It may be possible to rank the belief states themselves according to the entrenchment of beliefs contained within them. This ranking, in turn, can serve a very useful purpose in predicting actions or plans based on the entrenchment criteria discussed in section 2.6.

To demonstrate the role of entrenchment in predicting, we provide the following illustrative example. We may predict based on any criteria using a heuristic function which is suitable in the context of the problem.

The following are some heuristics which may be employed:

- The highest entrenchment value of a possible relation satisfied for each predictable action or plan.

  This criteria flows from satisfying the abstract notion of the epistemic entrenchment orderings amongst the fluents. It does not take into account the satisfaction of the postulate of informational economy.

  For example, the predictable relation 4.19 uses four fluents D_1 through to D_4 to predict the action D in state S. Now assume that the epistemic entrenchment ordering $D_1 \leq E_1$ exists, meaning that the action or plan E is more epistemically entrenched than the action or plan D. The *predictable* relation 4.20 of action E in state S, uses only two fluents E_1 and E_2. Since the entrenchment of action E is higher than that of D, this heuristic would prefer to predict E over D.

$$predictable(D, S) \Leftarrow$$

$$possible(D_1, S) \land$$

$$possible(D_2, S) \wedge$$
$$assumable(D_3, S) \wedge \tag{4.19}$$
$$assumable(D_4, S)$$

$$predictable(E, S) \Leftarrow$$
$$possible(E_1, S) \wedge$$
$$assumable(E_2, S) \wedge \tag{4.20}$$

- The highest entrenchment value of an assumable relation satisfied for each predictable action or plan.
- The sum of the epistemic entrenchment values of the relations comprising each predictable relation.
- The belief state with the highest epistemic entrenchment ordering. (A sum or product of the axioms contained within each knowledge base).

  The motivation for this heuristic lies in the fact that predictions made on the basis of such entrenchment measurements will be more probable, useful or reliable, than blind predictions [5, 14].

Chapter 5

Conclusion and Future Work

A plan recognition system must be able to use incomplete information to provide the required flexibility. A general plan recognition system must be able to perpetually accept revised descriptions of a world in which actions take place, including changes to the observed relationships amongst actions and plans. To anticipate the behaviour of agents acting in a dynamic observable world, the plan recognition system must be able to hypothesize consistent plans.

We use the Ghose-Goebel belief change model to maintain a dynamic set of assertions that represent actions and relationships amongst actions and plans. This method explicitly supports reasoning with incomplete knowledge and for revision of the belief base with newly observed facts. Observed actions are assimilated within the belief revision system and, together with current beliefs about actions and their affects, constrain the plans that can be recognized. Additionally, the Ghose-Goebel reasoning framework supports the maintenance of multiple mutually inconsistent states of the world, providing a basis to assume alternative hypothetical completions of plans.

The system we propose consists of two modules. The belief maintenance module utilizes the Ghose-Goebel Belief Revision System to maintain a body of knowledge based on observations. This system maintains a knowledge base and allows for its dynamic change based on new information observed by an agent. The knowledge base may be expanded, revised or contracted, depending on the type of new informa-

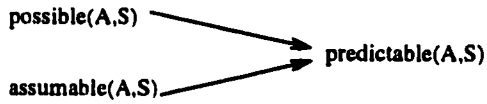


Figure 5.1: The two plan recognition relations.

tion which is observed. Multiple mutually inconsistent belief sets are maintained as potential belief states which that can be used to recognize plans.

The second module is responsible for plan recognition using the information about the world maintained by the first module. This system allows for observation of new information, certain types of predictions, and recognition of possible plans in the given state of the world. It relies on nonmonotonic assumptions which are used in the plan recognition and prediction aspects of its functionality.

We propose two fundamental plan recognition relations:

These are used to predict possible (meaning that they are executable) or assumable (meaning that they could be executable) plans and actions. Together with the multiply defined knowledge bases they provide a wide spectrum of plan recognition functionality.

Some of the drawbacks of our approach lie in the fact that the knowledge base which must be maintained will grow in size very quickly. This is due to a copy of a knowledge base which has to be maintained for each interpretation of a belief change operation.

This system is a proposal for a plan recognition system which presents a competent model which tries to maintain all of the possible information with which the system comes across. It is not implemented and only suggestions are provided as to the implementation process.

Not enough has been mentioned about the possible impact of entrenchment which beliefs may have. This information is crucial to determine the level of belief we may place in various pieces of information. This information in turn may be used by the system to prefer certain plans over some other plans when suggesting them to the user. If the user's preferences were taken into account in determining the entrenchment, the system would be user-tuned and not general, the way it is proposed now. This is something to consider in the future.

Another important future consideration involves the development of other relations and predicates which will allow for more complex relationships in the knowledge base to be retrieved. For example the relation $findSatisfyingAction(Action, Situation)$ might return the actions' names whose postconditions satisfy the specified action's preconditions in order to make it possible or assumable. Similarly

$$findSatisfyingPlan(Action, Situation) \tag{5.1}$$

could be used to find a sequence of actions which would have to be performed in order to satisfy the preconditions of the specified action. These are very powerful relations because they tell the user the sequence of actions or a single action which must be performed in order to perform another action.

Similarly, this idea may be extended to encompass all possible knowledge bases which exist as a result of belief change operations. These powerful relations would make it unnecessary to make a choice of a belief base used, and would be capable of predicting based on information in all of the knowledge bases.

Bibliography

[1] S. Carberry. *Pragmatic Modelling in Information System Interfaces*. PhD thesis, University of Delaware, Department of Computer and Information Sciences, 1985.

[2] S. Carberry. Modelling the user's plans and goals. *Computational Linguistics*, 14:23–27, 1988.

[3] E. Charniak and R. Goldman. A Bayesian model of plan recognition. *Artificial Intelligence*, 64:53–79, 1993.

[4] J.P. Delgrande and W.K. Jackson. Default logic revisited. In *Proceedings of the Second International Conference on Principles of Knowledge Representation*, pages 118–127, 1991.

[5] P. Gärdenfors. Belief revision: An introduction. In *Belief Revision*, pages 1–28. Cambridge University Press, 1992.

[6] A. Ghose. *Practical Belief Change*. PhD thesis, University of Alberta, Department of Computing Science, 1995.

[7] A. Ghose, P. Hadjinian, A. Sattar, J. You, and R. Goebel. Iterated belief change: a preliminary report. In *Proceedings of the Sixth Australian Joint Conference on Artificial Intelligence*, pages 39–44, Melbourne, Australia, November 17–19 1993.

[8] S. Hanks and D. McDermott. Default reasoning, nonmonotonic logics and the frame problem. In *Proceedings of AAAI-86*, pages 328–333, 1986.

[9] H. Kautz. *A Formal Theory of Plan Recognition.* PhD thesis, University of Rochester, Department of Computer Science, 1987.

[10] R. Kowalski. *Logic for Problem Solving.* Elsevier North Holland, Inc., 1979.

[11] V. Lifschitz. Formal theories of action. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, volume 2, pages 966–972, 1987.

[12] D. Lin. *Obvious Abduction.* PhD thesis, University of Alberta, Department of Computing Science, 1992.

[13] D. Lin and R. Goebel. A message passing algorithm for plan recognition. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, pages 280–285, 1991.

[14] A. Nayak, N. Foo, M. Pagnucco, and A. Sattar. Entrenchment kinematics 101. In *Proceedings of the 7th Australian Joint Conference on Artificial Intelligence*, pages 157–164, 1994.

[15] B. Nebel. *Reasoning and Revision in Hybrid Representation Systems.* Springer-Verlag, first edition, 1990.

[16] D. Poole. A logical framework for default reasoning. *Artificial Intelligence*, 36(1):27–47, 1988.

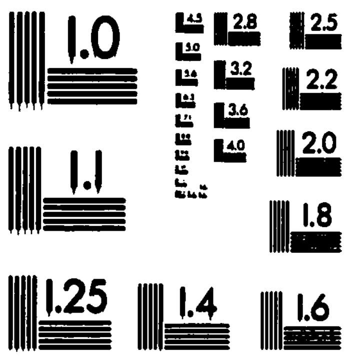
[17] D. Poole, R. Goebel, and R. Aleliunas. Theorist: A logical reasoning system for defaults and diagnosis. In N.J. Cercone and G. McCalla, editors, *The Knowledge Frontier: Essays in the Representation of Knowledge*, pages 331–352. Springer Verlag, New York, 1987.

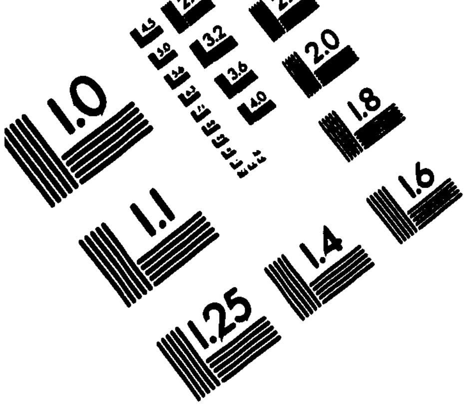
[18] R. Reiter. A logic for default reasoning. *Artficial Intelligence*, 13:81–132, 1980.

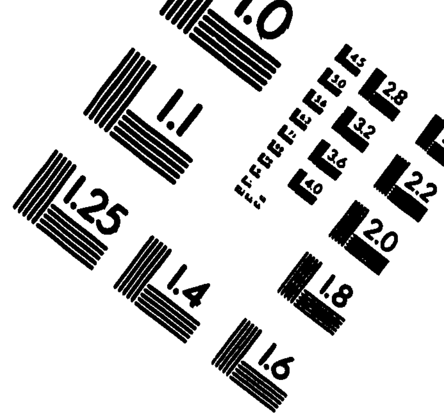
[19] A. Sattar. *Efficiency Aspects of Hypothetical Reasoning.* PhD thesis, University of Alberta, Department of Computing Science, 1992.

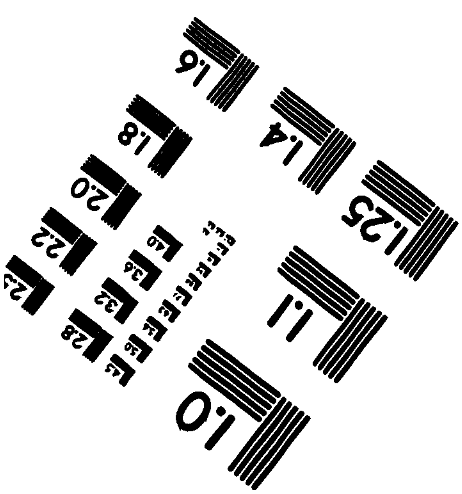
[20] P. van Beek. An investigation of probabilistic interpretations of heuristics in plan recognition. In *Proceedings of the Fifth International Conference on User Modelling*, pages 113–120, Kailua-Kona, Hawaii, 1996.

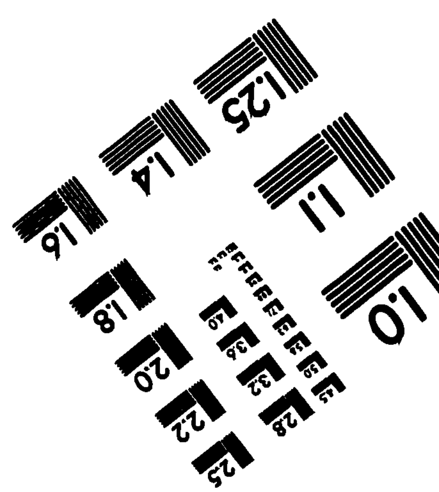
[21] M. Vilain. Getting serious about parsing plans: A grammatical analysis of plan recognition. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 190–197, 1990.











150mm

6″

APPLIED IMAGE . Inc

1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989