A General Abductive System with Application to Plan Recognition and Diagnosis

Hwee Tou Ng

June 1992 AI 92-177

A GENERAL ABDUCTIVE SYSTEM WITH APPLICATION TO PLAN RECOGNITION AND DIAGNOSIS

APPROVED BY

DISSERTATION COMMITTEE:

[illegible]

[illegible]

[illegible]

Bruce W. Porter

[illegible]

A GENERAL ABDUCTIVE SYSTEM WITH APPLICATION TO PLAN RECOGNITION AND DIAGNOSIS

by

HWEE TOU NG, B.Sc.,M.S.

DISSERTATION
Presented to the Faculty of the Graduate School of
The University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN
May, 1992

Acknowledgments

First, I would like to thank my advisor, Professor Raymond Mooney, for his invaluable guidance and support over the years. Ray recognized the importance of abduction and got me interested in this topic. He has set a consistently high standard of scholarly research and serves as an excellent role model for a computer scientist. He has been a great advisor, teacher, colleague, and friend.

I would also like to express my gratitude to Professors Woodrow Bledsoe, Benjamin Kuipers, Bruce Porter, and Robert Simmons, for serving on my thesis committee, and for their interest in my work. I am deeply indebted to Professor Kuipers who generously allowed me to use his Sun Sparc 2 workstation on which I made the final runs and collected the empirical results reported in this thesis. Ben had also provided me with travel support to attend the Sixth IEEE Conference on AI Applications in March 1990.

I am very grateful to Dr. Stanley Tuhrim of Mount Sinai School of Medicine, New York, and Dr. James Reggia of the University of Maryland at College Park who kindly provided me with the knowledge base on brain damage and the 50 patient test cases. Dr. Tuhrim answered my questions about the details of the knowledge base and test cases, for which I gratefully acknowledge.

Thanks to Adam Farquhar for kindly allowing me to use his ATMS code on which an earlier version of ACCEL was built, and for discussing technical details of the ATMS. Discussions with Siddarth Subramanian on diagnosis have been helpful. The Friday Explanation Group meeting has been a stimulating forum for discussion of AI research.

Thanks also go to Se June Hong, John Kastner, and Marshall Schor for giving me the opportunity to work at the IBM T. J. Watson Research Center in the summer of 1991, and for providing helpful feedback on my work.

This research was supported by a University of Texas MCD Fellowship (1987–89), an IBM Graduate Fellowship (1990–92), and by the NASA Ames Research Center under grant NCC-2-429. Computing equipment used was donated by Texas Instruments.

Last but certainly not least, my parents had taught me the value of hard work and laid a strong foundation in the early years of my education, without which I would not have come this far. Most of all, I would like to thank my wife, Boon Kheng, for her love, support, and encouragement that make life so wonderful.

Hwee Tou Ng

The University of Texas at Austin
May, 1992

A GENERAL ABDUCTIVE SYSTEM WITH APPLICATION TO PLAN RECOGNITION AND DIAGNOSIS

Publication No.______________

Hwee Tou Ng, Ph.D.
The University of Texas at Austin, 1992

Supervisor: Raymond J. Mooney

A diverse set of intelligent activities, including natural language understanding, diagnosis, and scientific theory formation, requires the ability to construct explanations for observed phenomena. In this thesis, we view explanation as *abduction*, where an abductive explanation is a consistent set of assumptions which, together with background knowledge, logically entails a set of observations.

To explore the practical feasibility of such a general abductive approach to explanation, we have successfully built a domain-independent system called ACCEL. In our system, knowledge about a variety of domains is uniformly encoded in first-order Horn-clause axioms. A general-purpose abduction algorithm, AAA, efficiently constructs explanations in the various domains by caching partial explanations to avoid redundant work. Empirical results show that caching of partial explanations can achieve more than an order of magnitude speedup in run time. We have applied our abductive system to two general tasks: plan recognition in text understanding, and diagnosis of medical diseases, logic circuits, and dynamic systems. The results indicate that ACCEL is a general-purpose system capable of plan recognition and diagnosis, yet efficient enough to be of practical utility.

In the plan recognition domain, we defined a novel evaluation criterion, called *explanatory coherence*, and tested ACCEL on 50 short narrative texts. Empirical results demonstrate that coherence is a better evaluation metric than simplicity in the plan recognition domain, and that our system is sufficiently general to be able to handle similar plan recognition problems not known to the system developer in advance.

In medical diagnosis, we prove that ACCEL computes the same diagnoses as the GSC model of Reggia, and present empirical results demonstrating the efficiency of ACCEL in diagnosing 50 real-world patient cases using a sizable knowledge base with over six hundred symptom-disease rules.

ACCEL also realizes model-based diagnosis, which concerns inferring faults from first principles given knowledge about the correct structure and behavior of a system. ACCEL has successfully diagnosed logic circuits (a full adder) and dynamic systems (a proportional temperature controller and the water balance system of the human kidney).

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Finding explanations for events and actions is an important aspect of general intelligent behavior. A diverse set of intelligent activities, including natural language understanding, diagnosis, scientific theory formation, and image interpretation, requires the ability to construct explanations for the phenomena observed. For instance, in text understanding, a reader infers the high-level goals and plans of the characters in a text in order to explain the events and actions described in the text. In dialog understanding, a participant infers the goals and plans of other participants based on the utterances exchanged in a conversation. We call this kind of inference *plan recognition*, which is known to be an important component of text and dialog understanding [Allen, 1987].

Similarly, in medical diagnosis, based on the observed symptoms of a patient, a physician infers the possible diseases that may explain the symptoms. In physical device diagnosis, based on the observed misbehavior of a physical device, a diagnostician infers the possible faults that may explain the misbehavior. In image interpretation, based on a two-dimensional image, a vision system infers the objects present in the scene that may explain the image.

In this thesis, we view explanation as *abduction*. The philosopher C.S. Peirce [Peirce, 1958] defined *abduction* as the process of finding the best explanation for a set of observations; i.e. inferring cause from effect. The standard logical formalization of abduction within artificial intelligence (AI) defines an abductive explanation as a consistent set of assumptions which, together with background knowledge, logically entails a set of observations [Charniak and McDermott, 1985]. Abduction has been proposed as a unifying formalism for explanation in a variety of tasks including natural language understanding, diagnosis, scientific theory formation, and image interpretation [Pople, 1973; Charniak and McDermott, 1985].

Formulating the generation of explanatory hypotheses as abduction has some advantages over the traditional "expert system" approach. In an expert system, heuristic rules of the form $e \rightarrow h$ are used to encode the fact that some evidence e may suggest hypothesis h. A separate inference engine deduces the set of possible hypotheses that are "implied" by the evidence. Conflict resolution strategies are employed to decide which rules should be fired first and hence to determine the most plausible hypotheses. Such an approach requires reversing the causal links between hypothesis and evidence. Also, control information about what to deduce is mixed with declarative knowledge about the relationship between hypothesis and evidence. This is in contrast to abduction, which encodes the relevant knowledge in its most natural form as "hypothesis $h \rightarrow$ evidence e". Abduction also relies on a separate evaluation criterion such as simplicity to determine which of the candidate hypotheses best explain some evidence. Abduction is therefore a more natural and declarative approach to modeling the generation of explanatory hypotheses.

While it has been realized for quite some time within AI that abduction is a general model for explanation, there have been no empirical explorations into the practical feasibility of such a general abductive approach to explanation. Many important questions remain unexplored. For example, is it possible to have a general-purpose yet efficient algorithm that can be used for making useful abductive inference in all the various domains? Do we need special-purpose control heuristics separately tailored for each domain? Do the criteria for selecting the best explanations vary according to the domain? How difficult is it to encode the knowledge necessary for constructing explanations in the various domains?

To address these important issues, we have successfully built a domain-independent system called ACCEL (Abductive Construction of Causal Explanations in Logic). In our system, knowledge about a variety of domains is uniformly encoded in first-order Horn-clause axioms. A general-purpose abduction algorithm, AAA (ATMS-based Abduction Algorithm), efficiently constructs explanations in these domains. We have applied our abductive system to two general tasks: plan recognition in text understanding, and diagnosis of medical diseases, logic circuits, and dynamic systems. We believe our approach represents a good trade-off between generality and efficiency — ACCEL is a general-purpose system capable of performing all of the above tasks, yet efficient enough to be of practical utility. In this thesis, we will present extensive empirical results demonstrating the efficacy and efficiency of our system in performing the above tasks.

Previous abduction algorithms and systems, when compared to ACCEL, are either too restrictive, too inefficient, or both. Although the ATMS algorithm of [de Kleer, 1986] has been proven to be a general abduction algorithm for propositional Horn-clause theories [Levesque, 1989], many interesting abduction tasks require the expressibility of first-order predicate logic. For example, the tasks of plan recognition in narrative texts, as well as abductive diagnosis of logic circuits and continuous dynamic systems, require that the domain theory be expressed in first-order predicate logic. Furthermore, in first-order logic, the important operation of unifying assumptions (factoring) becomes relevant. Frequently, simple and coherent explanations can only be constructed by unifying initially distinct assumptions so that the resulting combined assumption explains several observations [Pople, 1973; Stickel, 1988a]. This important problem does not arise in the propositional case.

On the other hand, the general-purpose first-order abduction algorithm proposed in [Stickel, 1988a] tends to perform a great deal of redundant work in that partial explanations are not cached and shared among multiple explanations. The ATMS algorithm, though it caches and reuses partial explanations in order to avoid redundant work, has not been extended to perform general first-order abduction. Also, the ATMS algorithm exhaustively computes all minimal explanations, which is computationally very expensive for large problems. Even in the propositional case, computing all minimal explanations is a provably exponential problem [McAllester, 1985; Selman and Levesque, 1990]. This indicates that resorting to heuristic search to find the best explanations is the most reasonable approach to building a practical abductive system.

Another important issue in abduction concerns the evaluation of the quality of explanations, i.e., what are the distinguishing features of a good explanation, and how can evaluation metrics be formulated so as to select and keep only the good explanations among the exponentially large number of explanations. Simplicity of explanations, defined as making the least number of assumptions in an abductive explanation, is a widely used metric to select the best explanations [Reggia *et al.*, 1983;

Charniak, 1986; Kautz and Allen, 1986]. In Chapter 3, we will give convincing evidence that simplicity is inadequate as an evaluation metric for explanations in text understanding.

Our algorithm AAA overcomes both the generality and efficiency problems in that it is an abduction algorithm for first-order Horn-clauses, and it uses ATMS-style caching to avoid redundant work. In Chapter 6, we will present empirical results which demonstrate that caching of partial explanations can achieve more than an order of magnitude speedup in run time. The AAA algorithm also incorporates a form of heuristic beam search in order to limit the computational efforts expended in finding the best explanations. Explanations are ranked according to their evaluation metric values and only the best explanations within the beam width of the beam search algorithm are kept.

Although ACCEL provides a more declarative approach to the generation of explanatory hypotheses than a traditional expert system, it is often necessary that axioms be formulated carefully so that the system will perform the desired task correctly and efficiently. As in traditional logic programming, it is frequently insufficient to just "state the correct knowledge" and expect the desired answers to be inferred. Appropriate programming methodologies must be developed so that a user knows how to axiomatize a problem to correctly and efficiently compute the desired answers [Poole, 1989a]. This is also true in "abductive logic programming". By successfully applying ACCEL to the tasks of plan recognition and diagnosis, we have demonstrated via many examples *how* a general abductive system can be used to achieve these tasks.

We now give a brief overview of the various domains on which ACCEL has been tested:

1. Plan recognition: We define a novel evaluation criterion, called *explanatory coherence*, and give empirical results demonstrating that coherence is a better evaluation metric than simplicity in plan recognition. We also give supporting evidence that our system is sufficiently general to be able to handle similar plan recognition problems not known to the system developer in advance.

2. Set covering diagnosis: We prove that, given the appropriate form of axioms, ACCEL computes the same diagnoses as those of the set-covering method of Reggia [Reggia *et al.*, 1983; Peng and Reggia, 1990]. We also present empirical results demonstrating the efficiency of ACCEL at diagnosing 50 real-world patient cases using a sizable knowledge base with over six hundred rules.

3. Model-based diagnosis: We use abduction to perform model-based diagnosis, which concerns inferring faults from first principles given knowledge about the correct structure and behavior of a system. The approach is applied to diagnosing logic circuits (a full adder) and dynamic systems (a proportional temperature controller and the water balance system of the human kidney). Empirical results are presented illustrating the capability of ACCEL in abductive diagnosis.

1.1 Organization of the Thesis

The rest of this thesis is organized as follows:

- Chapter 2 gives a formal definition of the abduction problem that we are addressing, and describes the AAA algorithm.
- Chapter 3 concerns abduction in the plan recognition domain.
- Chapter 4 concerns the use of general abduction to achieve diagnosis based on the set covering method.
- Chapter 5 presents ACCEL's abductive approach of model-based diagnosis.
- Chapter 6 presents empirical results on the speedup obtained through the use of caching in the AAA algorithm.
- Chapter 7 presents related work.
- Chapter 8 discusses future work.
- Chapter 9 gives the conclusion.
- Appendices A, B, and C list the knowledge base and test data used in the plan recognition, set covering, and model-based diagnosis domain, respectively.

Chapter 2

Problem Definition and Algorithms

2.1 Problem Definition

The abduction problem that we are addressing can be defined as follows.

Given:

- A set of universally quantified first-order Horn-clause axioms T (the domain theory), where an axiom is either of the form
  $C(v_1, \ldots, v_k) \leftarrow P_1(v_1, \ldots, v_k) \wedge \ldots \wedge P_r(v_1, \ldots, v_k)$ (a rule), or
  $F(v_1, \ldots, v_k)$ (a fact)
- An existentially quantified conjunction O of atoms (the input atoms) of the form
  $\exists v_1, \ldots, v_k\, O_1(v_1, \ldots, v_k) \wedge \ldots \wedge O_m(v_1, \ldots, v_k)$

Find:

All *explanations* with *minimal* (w.r.t. *variant-subset*) sets of assumptions.

We define *explanation*, *variant-subset*, and *minimality* as follows.

Definition 2.1 *Let A (the assumptions) be an existentially quantified conjunction of atoms of the form*

$$\exists v_1, \ldots, v_k\, A_1(v_1, \ldots, v_k) \wedge \ldots \wedge A_n(v_1, \ldots, v_k)$$

where $n \geq 0, A \cup T \models O$, and $A \cup T$ is consistent. An assumption set A (together with its corresponding proof) is referred to as an explanation *(or an* abductive proof*) of the input atoms.*

We will write A as the set $\{A_1, \ldots, A_n\}$ with the understanding that all variables in the set are existentially quantified and that the set denotes a conjunction.

Definition 2.2 *An assumption set A is a* variant-subset *of another assumption set B if there is a renaming substitution σ such that $A\sigma \subseteq B$.*

For example, $A = \{p(X, b)\}$ is a variant-subset of $B = \{p(Y, b), q(Y)\}$ with the renaming substitution $\sigma = \{X/Y\}$.[1]

[1] In this thesis, we denote variables by uppercase letters, and constants by lowercase letters.

Definition 2.3 *A set of explanations S is* minimal *if there is no explanation in S whose assumption set is a variant-subset of the assumption set of another explanation in S.*

Since the definition of abduction requires consistency of the assumed atoms with the domain theory, the abduction problem is in general undecidable. In our implemented system ACCEL, consistency checking is accomplished in two ways:

1. Using a pre-determined list of *nogoods*, where a nogood is a set of assumptions $\{A_1(v_1,\ldots,v_k),\ldots,A_n(v_1,\ldots,v_k)\}$ such that
$$\forall v_1,\ldots,v_k\ A_1(v_1,\ldots,v_k)\wedge\ldots\wedge A_n(v_1,\ldots,v_k)\rightarrow \mathit{false}.$$
Consistency checking ensures that an assumed set of atoms is not subsumed by any nogoods (i.e., no instance of a nogood is a subset of an assumed set of atoms);
2. Using procedural code to check for inconsistency of assumptions (for efficiency reasons).

2.2 The SAA Algorithm

Stickel has proposed an algorithm for computing the set of all first-order Horn-clause abductive proofs [Stickel, 1988a]. His algorithm, which we will call SAA (Stickel's Abduction Algorithm), operates by applying inference rules to generate goal clauses. The initial goal clause is the input atoms $O_1,\ldots,O_m$. Each atom in a goal clause can be marked with one of *proved, assumed*, or *unsolved.* All atoms in the initial goal clause are marked as unsolved. A final goal clause must consist entirely of proved or assumed atoms.

Let G be a goal clause $Q_1,\ldots,Q_n$, where the leftmost unsolved atom is Q_i. The algorithm SAA repeatedly applies the following inference rules to goal clauses G with unsolved atoms:

- *Resolution with a fact.* If Q_i and a fact F are unifiable with a most general unifier (mgu) σ, the goal clause $Q_1\sigma,\ldots,Q_n\sigma$ can be derived, where $Q_i\sigma$ is marked as proved.
- *Resolution with a rule.* Let $C \leftarrow P_1\wedge\ldots\wedge P_r$ be a rule where Q_i and C are unifiable with a mgu σ. Then the goal clause
$$Q_1\sigma,\ldots,Q_{i-1}\sigma,P_1\sigma,\ldots,P_r\sigma,Q_i\sigma,\ldots,Q_n\sigma$$
can be derived, where $Q_i\sigma$ is marked as proved and each $P_k\sigma$ is marked as unsolved.
- *Making an assumption.* If Q_i is *assumable*, then $Q_1,\ldots,Q_n$ can be derived with Q_i marked as assumed. By *assumable*, we mean that Q_i has been designated as an atom that the algorithm is allowed to assume. The algorithm also checks that all assumptions made are consistent with the domain theory.
- *Factoring with a proved or assumed atom.* If Q_j and Q_i $(j < i)$ are unifiable with a mgu σ, the goal clause
$$Q_1\sigma,\ldots,Q_{i-1}\sigma,Q_{i+1}\sigma,\ldots,Q_n\sigma$$
can be derived.

2.2.1 A Simple Example

To illustrate the working of the SAA algorithm, consider the following simple example. Suppose the knowledge base consists of the following axioms:

$p(X,Y) \leftarrow q(X,Z) \wedge r(Z,Y).$

$r(X,Y) \leftarrow s(Y) \wedge q(X,Y).$

$s(a).$

Suppose the input atom is $p(a,a)$. Then one possible sequence of inferences leading to an abductive proof for $p(a,a)$ with the assumption set $\{q(a,a)\}$ is shown in Figure 2.1.

2.2.2 Problems with the SAA Algorithm

The SAA algorithm as described above suffers from two problems.

Combinatorial Explosion It has been shown that, even in the propositional case, computing all minimal (w.r.t. subset) explanations is provably exponential [McAllester, 1985; Selman and Levesque, 1990], since in the worst case, the number of minimal explanations is exponentially large. The SAA algorithm computes all first order Horn-clause abductive explanations and therefore it is also at least an exponential algorithm. (Actually, since the SAA algorithm includes consistency checking of the assumptions, it is in general undecidable.)

However, in practice, what is needed is only the best explanation, or the best few explanations. To avoid combinatorially explosive computation, [Stickel, 1988a; Hobbs *et al.*, 1988] proposed the use of a cost metric to rank and heuristically search the more promising explanations first. Each input atom is assigned a cost of assuming that input atom. The antecedents $A_1, \ldots, A_n$ of every rule R in the knowledge base are assigned relative costs $C_1, \ldots, C_n$ so that when the algorithm backward-chains on a subgoal G using a rule R, the cost of each new antecedent subgoal A_i is $cost(G) \times C_i / \sum_{i=1}^{n} C_i$. The best explanation has assumptions with the least cumulative cost.

Simplicity, defined as making the minimum number of assumptions in an explanation (known as the *minimum* explanation), is another commonly used metric to select the best explanations [Reggia *et al.*, 1983; Charniak, 1986; Kautz and Allen, 1986]. However, previous work has shown that finding the minimum abductive explanation is NP-hard [Reggia *et al.*, 1983; Reggia *et al.*, 1985; Allemang *et al.*, 1987; Bylander *et al.*, 1989].

In this thesis, we used a form of beam search to overcome the computational intractability problem. Evaluation metrics including a coherence metric and a simplicity metric are used to determine the quality of an abductive proof, and a limited list of the best abductive proofs are maintained during the search.

Redundant Inference Even with the use of heuristic search to restrict the computation expended in finding the good explanations, the SAA algorithm can still perform a great deal of redundant work in that partial explanations are not cached and shared among multiple explanations. To see why this is the case, consider the two examples shown in Figures 2.2 and 2.3.

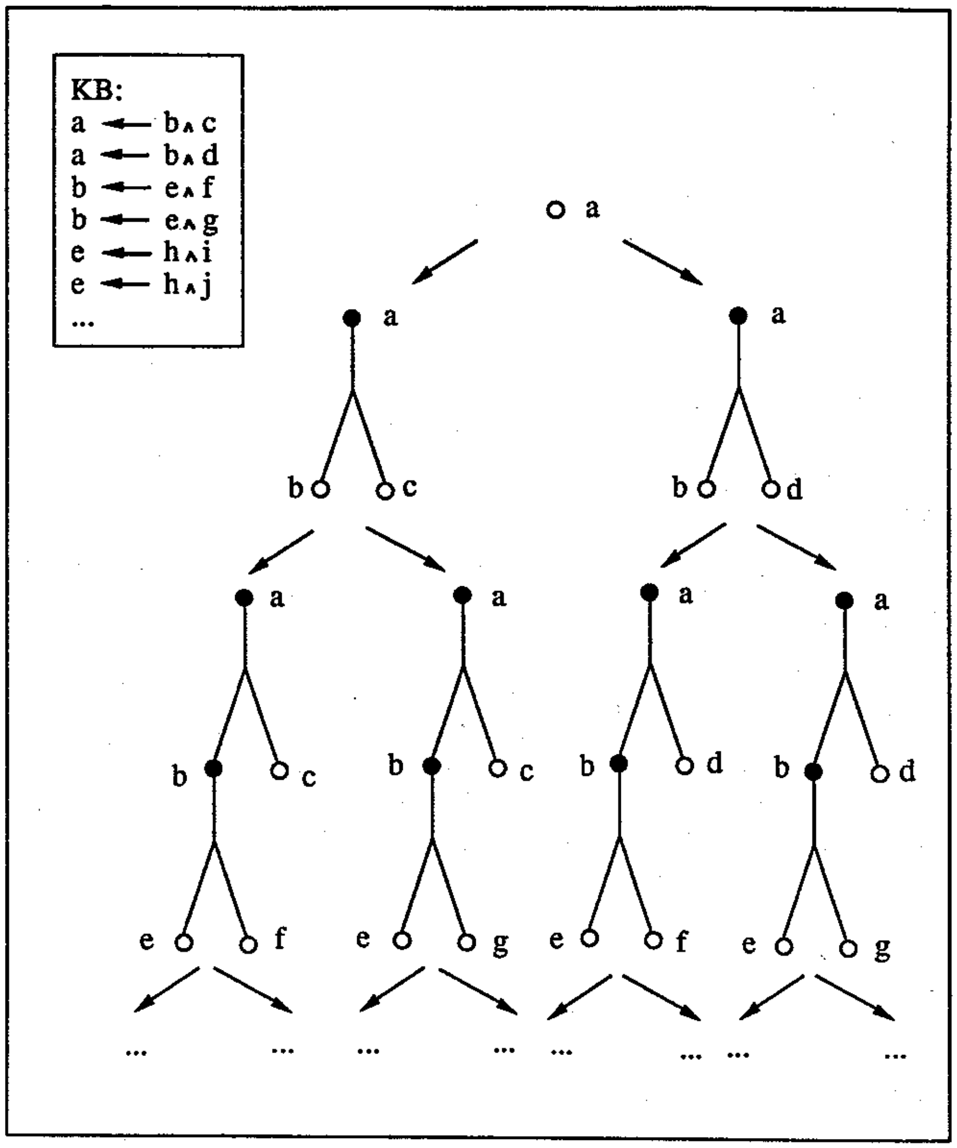

Figure 2.3: Duplicating inference: example 2.

In the first example, after backward-chaining on the rule $a \leftarrow b \wedge c \wedge d$, the SAA algorithm can either make b an assumption, or backward-chain on the rule $b \leftarrow e \wedge f$. This introduces two partial abductive proofs, both have the identical subgoals c and d. The SAA algorithm will then duplicate the same inferences in expanding the subgoals c and d in the two partial proofs. Since the proof tree rooted at the subgoals c and d can be arbitrarily deep, substantial effort will be wasted duplicating inferences.

In the second example, each time a subgoal (like a, b, and e) is expanded by backward-chaining, two partial proofs are generated, and inferences will be duplicated across the two successor partial proofs. For instance, the rule $b \leftarrow e \wedge f$ is applied twice, the rule $e \leftarrow h \wedge i$ is applied four times, etc.

Note that this problem of duplicating inference arises in deductive theorem proving too. However, we believe that duplicating inference poses a more serious problem in abduction because multiple abductive proofs must usually be pursued in the search for a best explanation, whereas in deduction, we are usually interested in a single deductive proof. The need for multiple abductive proofs tends to result in more duplicate inferences being made. Also, note that the situation in example 1 arises each time an assumption is made, and since making assumptions occurs frequently in abduction, duplicating inferences almost always arise in abduction. In Chapter 6, we present empirical results showing that avoiding duplicate inferences can achieve more than an order of magnitude speedup.

2.2.3 Variant-subsets

The explanations generated by the SAA algorithm may include some that are variant-subsets of another. For instance, given the following axioms:

$$inst(G, going) \leftarrow inst(S, shopping) \wedge go\text{-}step(S, G).$$

$$goer(G, P) \leftarrow inst(S, shopping) \wedge go\text{-}step(S, G) \wedge shopper(S, P).$$

and the input atoms $inst(go1, going)$ and $goer(go1, john1)$, we can derive the explanation F with assumptions $A_F = \{inst(X, shopping), go\text{-}step(X, go1), inst(Y, shopping), go\text{-}step(Y, go1), shopper(Y, john1)\}$ by backward-chaining on the two axioms. Applying the factoring operation, we can obtain another explanation E with assumptions $A_E = \{inst(X, shopping), go\text{-}step(X, go1), shopper(X, john1)\}$. But note that although $A_E \not\subseteq A_F$, $A_E\sigma \subset A_F$ with the renaming substitution $\sigma = \{X/Y\}$.

Note that the variant-subset relation is a special case of subsumption. A subsumes B if $A\sigma \subseteq B$ for some substitution σ. However, for A to be a variant-subset of B, the substitution σ must be a renaming substitution.

Since explanations that are variant-supersets of other explanations are essentially redundant, they need to be eliminated. Unfortunately, it can be readily shown that determining variant-subset relation is an NP-complete problem by reduction from directed subgraph isomorphism, a known NP-complete problem [Garey and Johnson, 1979, page 202]. A directed graph $G = \langle V, E \rangle$ is transformed into an assumption set A as follows: for every vertex $v \in V$, add the assumption $N(X_v)$ to A, where X_v is a variable; for every directed edge $e = (v_1, v_2) \in E$, add the assumption $E(X_{v_1}, X_{v_2})$ to A, where X_{v_1}, X_{v_2} are variables. That is, $|A| = |V| + |E|$. It follows that G_1 is isomorphic to a subgraph of G_2 if and only if A_1 is a variant-subset of A_2. Hence, determining variant-subsets introduces yet another source of computational complexity when finding the minimal explanations in a first-order Horn-clause theory.

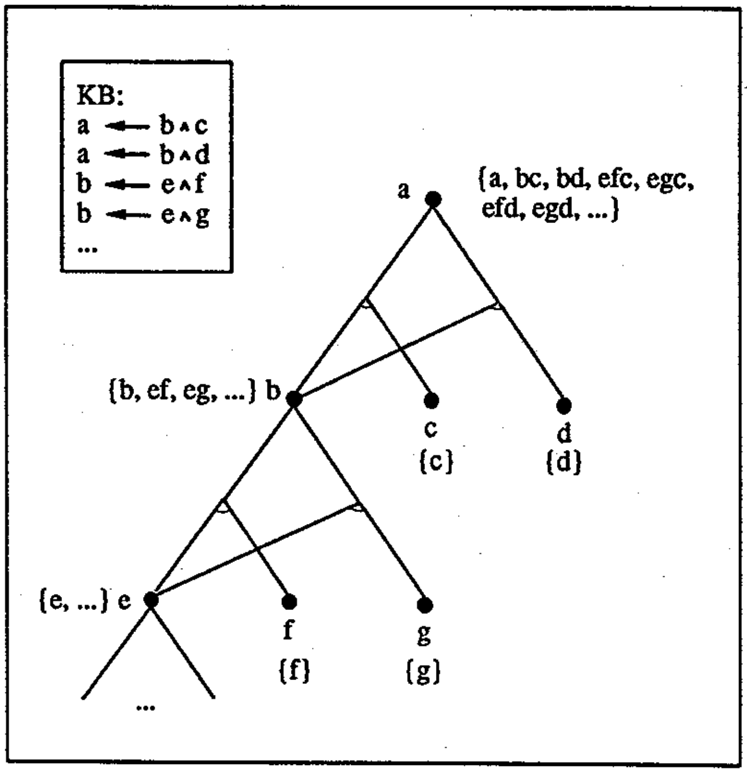

Figure 2.4: Caching and sharing inference steps.

2.3 The AAA Algorithm

We now present the abduction algorithm used in Accel, called AAA (ATMS-based Abduction Algorithm). This algorithm is much like the SAA algorithm, except that the abductive proofs for a subgoal G are cached and reused when the subgoal G or an instance of G is encountered subsequently in the search for the best abductive proofs.

To illustrate the idea of proof caching in AAA, consider the example shown in Figure 2.3. In the AAA algorithm, each of the rules $b \leftarrow e \wedge f$, $b \leftarrow e \wedge g$, $e \leftarrow h \wedge i$, etc., will be applied only once. Associated with each proved subgoal, we store all the abductive proofs of that subgoal. Figure 2.4 shows the abductive proofs associated with each subgoal in finding the abductive proofs for a in the AAA algorithm.

When the subgoal b is first encountered as an antecedent in the rule $a \leftarrow b \wedge c$, its abductive proofs are constructed by backward-chaining on the rules $b \leftarrow e \wedge f$, $b \leftarrow e \wedge g$, etc., in the knowledge base in a depth-first search order. When all the abductive_proofs of b are found, they are associated with b and stored in a cache. Subsequently, backward-chaining using the second rule $a \leftarrow b \wedge d$ will encounter the subgoal b again. At this time, all the previously found abductive proofs of b will be reused without recomputing them again.

2.3.1 Definition and Algorithm

The use of proof caching is very much in the style of an Assumption-based Truth Maintenance System (ATMS) [de Kleer, 1986], which caches and reuses partial explanations to avoid redundant work. The ATMS is a general facility for managing logical relationships among propositional formulas. It maintains multiple contexts at once and is particularly suitable for problem solving that involves constructing and comparing multiple explanations. In the ATMS, each problem solving datum is associated with a *node*. A node in the ATMS can be further designated as an *assumption*. Nodes are related via *justifications*. A justification is a propositional Horn-clause of the form $A_1 \wedge \ldots \wedge A_n \rightarrow C$, where each A_i is an antecedent node and C is the consequent node. An *environment* is a set of assumptions. Associated with each node is a set of environments called its *label*. The difference between AAA and the ATMS is that AAA constructs first-order Horn-clause proofs while the ATMS only deals with propositional Horn-clauses. Also, AAA is a backward-chaining algorithm while the traditional ATMS is forward-chaining.

We now give a formal description of the AAA algorithm. We will use similar terminology as in the ATMS.

Definition 2.4 *Let $E = \langle A, \sigma \rangle$, where A is a set of assumptions $\{A_1(v_1, \ldots, v_k), \ldots, A_n(v_1, \ldots, v_k)\}$ to be interpreted as an existentially quantified conjunction of assumptions, and σ is a substitution. We say that an atom G has an* environment *E iff $A \cup T \models G\sigma$ and $A \cup T$ is consistent.*

Note that a substitution σ is included as part of an environment E. This allows us to know directly from an environment $E = \langle A, \sigma \rangle$ associated with an atom G which instance of G is provable from the assumptions A.

Definition 2.5 *The label of an atom G (denoted label(G)) is a set of environments $\{E_1, \ldots, E_n\}$, to be interpreted as a disjunction of environments $E_1 \vee \ldots \vee E_n$, where $E_i = \langle A_i, \sigma_i \rangle$, such that*

- *(Soundness) $A_i \cup T \models G\sigma_i$;*
- *(Consistency) $A_i \cup T$ is consistent;*
- *(Completeness) For any consistent set of assumptions A where $A \cup T \models G$, A is a subsumed by some A_i; and*
- *(Minimality) No A_i is a variant-subset of some other A_j.*

The label of an atom is thus the set of all minimal (w.r.t. variant-subset) explanations of the atom G. Note that the set of explanations in a label is minimal w.r.t. to variant-subset but not subsumption. This is because a set of assumptions A that is obtained by factoring another set of assumptions B is such that A is subsumed by B (since $A = B\sigma$ for some substitution σ), but we do not want to remove A from the label since factoring frequently results in better explanations.

Without loss of generality, we can assume that the task of abduction is to find all minimal explanations of an atom $O(v_1, \ldots, v_k)$, since all explanations of

```
compute-label($G$,$D$)
if cache-lookup($G$,$D$) succeeds then return
label($G$) $\leftarrow \emptyset$
if $G$ is assumable then
  label($G$) $\leftarrow \{\langle\{G\},\{\}\rangle\}$
for each fact $F$ unifiable with $G$
    rename the variables in $F$
    $\sigma \leftarrow$ mgu($F,G$)
    label($G$) $\leftarrow$ label($G$) $\cup\{\langle\emptyset,\sigma\rangle\}$
backward-chain($G$,$D$)
store ($G$,$D$,label($G$)) in the cache
```

Table 2.1: The AAA Algorithm: compute-label.

$O_1(v_1,\ldots,v_k) \wedge \ldots \wedge O_m(v_1,\ldots,v_k)$ is the same as all explanations of $O(v_1,\ldots,v_k)$ once we add the rule

$$O(v_1,\ldots,v_k) \leftarrow O_1(v_1,\ldots,v_k) \wedge \ldots \wedge O_m(v_1,\ldots,v_k)$$

to the domain theory.

The AAA algorithm is presented in Tables 2.1–2.5. The top level procedure is *compute-label*(G,D), which will compute and return all possible abductive proofs of depth D or less of the atom G. In order to limit the search to the promising explanations, the algorithm will only maintain at most β_{intra} number of best explanations for each subgoal encountered in the search, where the quality of an explanation is determined by some evaluation metric (such as coherence or simplicity). When $\beta_{intra} = \infty$, all possible abductive proofs of depth D are computed. β_{intra} is thus the beam width of the heuristic beam search used to limit the computational efforts expended in finding the best explanations. We used beam search instead of best-first search since best-first search requires maintaining the complete list of partial explanations and so is too memory-consuming. If we let $\beta_{intra} = 1$, the beam search algorithm becomes a hill-climbing algorithm.

The AAA algorithm presented in this thesis supersedes a previous version reported in [Ng and Mooney, 1991b; Ng and Mooney, 1991a] which is incomplete. Specifically, the previous version misses explanations that are obtained when, during resolution of a subgoal with a fact or the consequent of a rule, the most general unifier is such that some variables in the subgoal are instantiated.

2.3.2 Indexing and Cache Lookup

The label of a subgoal, once computed, is stored in the cache indexed under the subgoal. The cache indexing scheme implemented in ACCEL is discrimination tree indexing, as described in [Norvig, 1992].

When AAA queries the cache for some subgoal G, if there exists in the cache some previous subgoal G' that is an alphabetic variant of G or is more general than G (i.e., $G = G'\sigma$ for some substitution σ), then the appropriate subset of G''s

```
backward-chain($G$,$D$)
if $D = 0$ then return
for each rule $C \leftarrow P_1 \wedge \ldots \wedge P_r$
    rename the variables in the rule
    $\sigma \leftarrow$ unify($C$,$G$)
    if $\sigma \neq$ fail then
      $P_1 \leftarrow P_1\sigma, \ldots, P_r \leftarrow P_r\sigma$
      compute-label($P_1, D-1$)
      partial-envs $\leftarrow \emptyset$
      for each environment $\langle A, \lambda \rangle \in$ label($P_1$)
          partial-envs $\leftarrow$ partial-envs $\cup \{\langle A, \sigma\lambda \rangle\}$
      for each $P_i \in \{P_2, \ldots, P_r\}$
          partial-envs $\leftarrow$ cross-product(partial-envs,$P_i$,$D$)
          variant-subset-minimize(partial-envs)
          if |partial-envs| $> \beta_{intra}$ then
            sort partial-envs by the evaluation metric
            truncate the size of partial-envs to $\beta_{intra}$
      label($G$) $\leftarrow$ label($G$) $\cup$ partial-envs
variant-subset-minimize(label($G$))
if |label($G$)| $> \beta_{intra}$ then
  sort label($G$) by the evaluation metric
  truncate the size of label($G$) to $\beta_{intra}$
```

Table 2.2: The AAA Algorithm: backward-chain.

```
cross-product(partial-envs,$P$,$D$)
old-partial-envs $\leftarrow$ partial-envs
partial-envs $\leftarrow \emptyset$
for each $E_1 = \langle A_1, \sigma_1 \rangle \in$ old-partial-envs
    compute-label($P\sigma_1, D-1$)
    for each $E_2 = \langle A_2, \sigma_2 \rangle \in$ label($P\sigma_1$)
        $E \leftarrow \langle A_1\sigma_2 \cup A_2, \sigma_1\sigma_2 \rangle$
        if $E$ is consistent then
          partial-envs $\leftarrow$ partial-envs $\cup$ factoring($E$,$A_1\sigma_2$,$A_2$)
return(partial-envs)
```

Table 2.3: The AAA Algorithm: cross-product.

$$A_4 : \langle\{q(b,V), s(V,b), t(V)\}, \{Y/V, X/b\}\rangle$$

The second environment E_2 of the label of $q(X,Y)$ has the substitution $\{X/a, Y/Z\}$, so another recursive call *compute-label*$(r(Z,a),\infty)$ is made. However, since the more general subgoal $r(Y,X)$ has been encountered previously and its abductive proofs are cached, AAA reuses the appropriate subset of the label of $r(Y,X)$ in the cache. The environments returned from the call *compute-label*$(r(Z,a),\infty)$ are (after renaming) $G_1 = \langle\{r(Y',a)\}, \{Z/Y', X'/a\}\rangle$ and $G_2 = \langle\{s(U',a), t(U')\}, \{Z/U'\}\rangle$. Taking the union of the appropriately instantiated environment E_2 and the environments G_1 and G_2 yields two additional abductive explanations for the label of $p(X)$:

$$A_5 : \langle\{s(a,Y'), r(Y',a)\}, \{X/a, Y/Y', Z/Y', X'/a\}\rangle$$

$$A_6 : \langle\{s(a,U'), s(U',a), t(U')\}, \{X/a, Y/U', Z/U'\}\rangle$$

Finally, factoring of the assumptions $s(a,U')$ and $s(U',a)$ in A_6 yields an additional abductive explanation for the label of $p(X)$:

$$A_7 : \langle\{s(a,a), t(a)\}, \{X/a, Y/a, Z/a, U'/a\}\rangle$$

The label of $p(X)$ computed is $\{A_1, \ldots, A_7\}$.

2.3.4 Enhancements to AAA

The AAA algorithm presented above is actually a slight simplification of the one implemented in ACCEL. For the sake of clarity in exposition, we have omitted some less essential details of the algorithm in Tables 2.1–2.5. We now describe the enhancements to the basic algorithm to make it more useful and efficient:

- As described above, an environment only has a set of assumptions and a substitution, which gives the set of assumptions sufficient to prove the instantiated atom using the domain theory. In plan recognition for natural language stories, the evaluation metric used, coherence, requires the structure of the proof in order for the coherence metric to be computed. As such, an environment is extended to include also the rules needed to prove the atom, in addition to the assumptions. These rules are collected as the algorithm backward-chains through the rules in the domain theory.

- In addition to D and β_{intra}, the AAA algorithm has a few additional parameters so that the algorithm can be specialized to execute more efficiently in each of the different domains.

    1. β_{inter}: This parameter controls the number of explanations kept after processing each of the input atoms in the conjunction $O_1 \wedge \ldots \wedge O_m$ given to ACCEL. It is always the case that $\beta_{inter} \leq \beta_{intra}$, since β_{intra} determines the number of explanations kept at every subgoal, and so the number of explanations kept at an input atom can be no more than β_{intra}.

    2. Factoring: This is a parameter to control if factoring should be performed. Factoring is essential in the plan recognition domain and the set-covering diagnosis domain, but it is turned off in the model-based diagnosis domain. The reason for the choices will be explained in subsequent chapters.

3. Variant-superset: Since eliminating variant-supersets is an expensive operation, there is a parameter to control whether it should be used. Eliminating variant-supersets is necessary in the plan recognition domain, but it is turned off in the model-based diagnosis domain (the reason will be explained in subsequent chapters). It is specialized to removing (simple) supersets in the set-covering domain, since the set-covering domain is propositional.

4. Evaluation metric: Each domain has its own explanation evaluation metric to determine the quality of a given explanation. In the plan recognition domain, coherence is a better evaluation metric, whereas in the diagnosis domains, the simplicity metric suffices.

5. Assumable predicates: The atoms that are assumable vary according to the domain. In the plan recognition domain, all atoms are assumable. In the diagnosis domain, only atoms corresponding to diseases or behavioral modes (normality, fault modes, abnormality) are assumable. Abduction in which the assumable atoms are restricted to a pre-determined set of predicates is known as *predicate specific abduction* [Stickel, 1988a].

- Taking the cross product of two labels L_1 and L_2 (i.e., for each environment $E_i \in L_1$ and each environment $E_j \in L_2$, form the environment $E_i \cup E_j$) is a very expensive operation and so the algorithm tries to minimize generating at once all the possible combined environments in the cross product of two labels. It accomplishes this by heuristically estimating the goodness of the two individual environments and their combined environment without first generating the complete combined environment. More details on this will be given in subsequent chapters on the various domains.

- In the procedure ***backward-chain***, the algorithm skips to the next rule if partial-envs $= \emptyset$ at any point while taking cross product of the labels of the antecedents of a rule $C \leftarrow P_1 \wedge \ldots \wedge P_r$. This is because when the computed partial-envs for a subset of the antecedents $P_1, \ldots, P_i$ is empty, then regardless of the labels of $P_{i+1}, \ldots, P_r$, partial-envs for the entire conjunction of the antecedents $P_1 \wedge \ldots \wedge P_r$ will still be empty.

- Before expanding a subgoal G (i.e., making it an assumption, or resolving it with facts or rules), first check that G is consistent with the input atoms $O_1, \ldots, O_i$ processed so far, since if G is inconsistent with $O_1, \ldots, O_i$, then its explanations are of no interest in the context of finding abductive proofs for the input atoms $O_1 \wedge \ldots \wedge O_m$.

Chapter 3

Plan Recognition

3.1 Introduction

According to Charniak and McDermott [Charniak and McDermott, 1985], "the principal component of understanding is the extraction of as many correct and useful inferences as possible from an utterance or text." A particularly useful form of inference is plan recognition — determining the goals and plans of characters in a narrative or participants in a dialog based on their observed actions or utterances. Plan recognition has proven to be a critical aspect of generating appropriate responses to questions as well as resolving ambiguities and anaphora [Allen, 1987].

In this thesis, we model plan recognition as abduction. Natural language understanding has recently been studied in terms of abduction. Specifically, abduction has been used to solve problems ranging from resolving anaphoric references and syntactic ambiguity [Hobbs *et al.*, 1988] to resolving lexical ambiguity and recognizing characters' plans in a narrative text [Charniak, 1986; Charniak and Goldman, 1991].

We do not focus on the parsing aspect of natural language understanding, and ACCEL does not accept natural language input. Instead, we assume the existence of some appropriate parser that translates a given set of input sentences into a logical representation consisting of an existentially quantified conjunction of input atoms. Given such a logical representation of the literal meaning of a narrative text, ACCEL infers an "embellished" interpretation by constructing an abductive proof in which a set of higher-level plans is assumed and the assumed plans logically entail the characters' observed actions. An abductive proof is considered an interpretation of the input sentences.

We will present a novel evaluation criterion, called *explanatory coherence*, for evaluating abductive explanations in the plan recognition domain [Ng and Mooney, 1990]. We have tested ACCEL on a set of 50 short narrative texts. The first 25 narrative texts were taken from Goldman's thesis [Goldman, 1990] in which they were used to test a probabilistic approach to natural language understanding. After the knowledge base has been developed to successfully process the first 25 texts, an additional set of 25 narrative texts unknown to the system developer is used to further evaluate the performance of our system. We will also present empirical results showing that our coherence metric is a better measure for selecting the best explanation in the plan recognition domain than an alternative approach based on simplicity. It also performs as well as another approach based on probability, but without requiring elaborate engineering of the appropriate probabilities.

3.2 Explanatory Coherence

3.2.1 Motivation

In previous research on abduction for text understanding and plan recognition, simplicity has been proposed as a metric for selecting the best explanation. For instance, in [Charniak, 1986], the best interpretation is one that maximizes $E - A$, where $E =$ the number of explained observations, and $A =$ the number of assumptions made. The work of Kautz explicitly incorporates the assumption of minimizing the number of top-level events in deducing the plan that an agent is pursuing [Kautz and Allen, 1986].

Though an important factor, the simplicity criterion is not sufficient by itself to select the best explanation. In the area of language understanding, we argue that some notion of explanatory coherence is more important in deciding which explanation is the best. Consider the sentences: "Mary had a heart attack. John is depressed." The sentences translate into the conjunction of the following atoms: name(m,mary), has(m,h), heart-attack(h), name(j,john), and depressed(j). A knowledge base of axioms relevant to these input atoms are:

depressed(X) ← like(X,Y) ∧ bad(condition(Y)) ∧ irreplaceable(Y)

depressed(X) ← pessimist(X)

bad(condition(X)) ← has(X,Y) ∧ illness(Y)

illness(X) ← heart-attack(X)

Based on the above axioms, there are two possible interpretations of these sentences, as shown in Figure 3.1. Suppose the simplicity metric is defined as the inverse of the number of assumptions made, where every leaf node in the proof graph counts as an assumption, including input atoms that are not explained (the assumptions in Figure 3.1 are underlined). Relying on this simplicity metric results in selecting the interpretation that John is depressed because he is a pessimist, someone who always feels gloomy about life (Figure 3.1b). This is in contrast to our preferred interpretation of the sentences — John is depressed because John likes Mary and Mary had a heart attack (Figure 3.1a).

Note that varying the definition of simplicity somewhat will not help here. For instance, using the simplicity criterion of [Reiter, 1987] based on subset minimality does not work well for this example — it is indifferent towards both interpretations, instead of choosing the preferred one. If we decide not to count input atoms as assumptions, then the preferred interpretation still makes more assumptions (four) compared to only one assumption in the other interpretation. Charniak's simplicity metric of $E - A$ also will not work, since both explanations would explain exactly one input atom, that John is depressed.

Intuitively, it seems that the first interpretation (Figure 3.1a) is better because the input atoms are connected more "coherently" than in the second interpretation (Figure 3.1b). We manage to connect "John is depressed" with "Mary had a heart attack" in the first interpretation, whereas in the second interpretation, they are totally unrelated. This is the intuitive notion of what we mean by *explanatory coherence*, i.e., how well the various parts of the input sentences are "tied together" in the interpretation.

That sentences in a natural language text are connected in a coherent way is reflected in the well known "Grice's conversational maxims" [Grice, 1975], which are principles governing the production of natural language utterances, such as "be

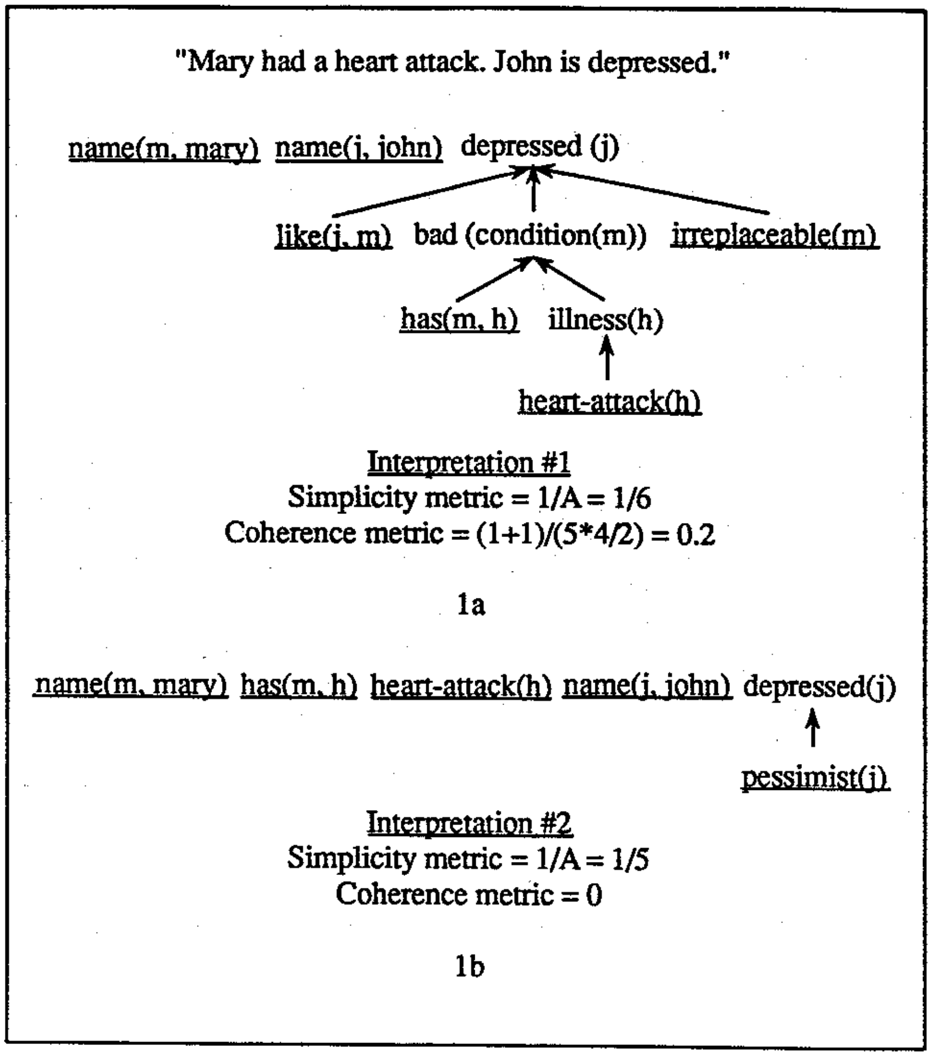

Figure 3.1: The importance of explanatory coherence.

relevant", "be informative", etc. Although the notion that natural language text is coherently structured has long been recognized by researchers in natural language processing (see [Allen, 1987]), our work is the first to incorporate the notion of coherence in the context of evaluating an abductive explanation.

3.2.2 Definition

We would like to formulate our coherence metric so as to possess several desirable properties. In particular, explanations with more connections between any pair of input atoms should have higher coherence metric values. Also, a coherence metric with values lying in a unit range 0–1 will facilitate the comparison of explanations. We have developed a formal characterization of what we mean by explanatory coherence in the form of a coherence metric satisfying these properties.

Definition 3.1 *The coherence metric C is defined as follows:*

$$C = \frac{\sum_{1 \leq i < j \leq l} N_{i,j}}{l(l-1)/2}$$

where

l = the total number of input atoms;

$N_{i,j}$ = 1 if there is some node n in the proof graph such that there is a (possibly empty) sequence of directed edges from n to n_i and a (possibly empty) sequence of directed edges from n to n_j, where n_i and n_j are input atoms. Otherwise, $N_{i,j}$ = 0.

The numerator of this metric is the total number of pairs of input atoms that are connected. The denominator of the metric scales the sum according to the size of the explanation so that the final metric value falls between 0 and 1. Note that the definition of coherence given in this thesis is a slight modification of the one given in [Ng and Mooney, 1990; Ng and Mooney, 1989]. The new definition remedies the anomaly reported in [Norvig and Wilensky, 1990] of occasionally preferring spurious interpretations of greater depths.

To illustrate the computation of the coherence metric, consider the explanation in Figure 3.1a. Let n_1 = (name m mary), n_2 = (name j john), n_3 = (depressed j), n_4 = (has m h), and n_5 = (heart-attack h). The total number of input atoms l = 5. In this explanation, $N_{3,4} = 1$, since there is a node n_4 such that there is a directed path from n_4 to n_3 and also a directed path from n_4 to n_4 (the trivial empty path). Similarly, $N_{3,5} = 1$. All other $N_{i,j} = 0$. This results in the coherence metric $C = 0.2$, as shown in Figure 3.1a.

The coherence metric C has the following properties:

1. $0 \leq C \leq 1$. $C = 0$ when the input atoms are totally unrelated. For instance, $C = 0$ in the interpretation of Figure 3.1b. $C = 1$ if the proof graph is completely connected, i.e., for each pair of input atoms, there is some node connecting both input atoms.

2. C attempts to take into account pairwise connectivity between any two input atoms. Thus, a proof graph that is split into several partitions will have low coherence value, since any two input atoms n_i and n_j from two disjoint partitions of the proof graph have $N_{i,j} = 0$.

We note here some advantages of our coherence metric:

1. Coherent explanations are often simple explanations. This is because in a coherent explanation, propositions tend to be more tightly connected together. This increases the likelihood of assumptions being unified, and leads to a reduction in the number of assumptions made and thus a simpler explanation.

2. Compared to the simplicity metric, the coherence metric is less vulnerable to changes in the underlying representation of the knowledge base. It is relatively easy to encode the axioms in a knowledge base in a slightly different way so as to change the number of assumptions made in an explanation. However, connections between propositions are less dependent (relatively speaking) on such changes. For example, suppose we change the axioms in the given example slightly so that as long as a person likes something that is in a bad condition (but not necessarily irreplaceable), then that person is depressed. Also, suppose one has to be poor as well as a pessimist to be depressed. Given this modified set of axioms, the first interpretation now requires only five assumptions, while the second interpretation requires six. So all of a sudden, the first interpretation becomes the simpler explanation of the two. However, the coherence metric values of both interpretations remain unchanged.

3. Evaluating explanations based on coherence also nicely resolves a problem in abduction, that of deciding on the appropriate level of specificity of explanations. Previous approaches fall into several categories: most specific abduction, least specific abduction, cost-based (weighted) abduction, and predicate specific abduction. In most specific abduction, the assumptions made must be *basic*, i.e., they cannot be "intermediate" assumptions that are themselves provable by assuming some other (more basic) assumptions [Cox and Pietrzykowski, 1987]. In least specific abduction, the only allowable assumptions are the input atoms [Stickel, 1988a]. In cost-based abduction, costs (or weights) are assigned to the antecedents of backward-chaining rules in order to influence the decision on whether to backward-chain on a rule [Hobbs *et al.*, 1988]. In predicate specific abduction, the assumptions made must have predicates from a pre-determined set of predicates.

   However, none of the above approaches is completely satisfactory. Least specific abduction is too restrictive since frequently, assumptions other than the input atoms must be made, such as those to be inferred by a reader. Most specific abduction is also too rigid since it is not always the case that we want to explain everything in terms of every available cause, since the causes that explain different input atoms may be completely unrelated to one another. Cost-based abduction could presumably arrive at the correct explanation given the "appropriate" set of costs, but it is unclear how the costs can be assigned in general to work on all problems. Predicate specific abduction is not suitable for text understanding since the assumptions made by a reader in text understanding are not restricted to a fixed set of predicates.

   In our approach, the desired specificity of an explanation is one which maximizes coherence. That is, we backward-chain on rules to prove the subgoals in an explanation only if doing so increases its overall coherence, and thus we make assumptions just specific enough to connect the input atoms. Coherence has been successfully used to determine the appropriate level of specificity of explanations for the 50 narrative texts processed by ACCEL. ~~Hence~~, we believe our coherence-based approach is better than the alternative approaches in determining the specificity of explanations.

Finally, we want to point out that it is *not* our belief that simplicity is completely irrelevant to the selection of explanations. (In fact, in the diagnosis domain, we rely on simplicity as our evaluation metric.) Rather, we consider explanatory coherence to be a *more* important criterion in selecting good explanations in the domain of text understanding and plan recognition. As such, we evaluate explanations in the plan recognition domain based on their coherence. When there is a tie between the coherence metric values of two explanations, we then rely on the simplicity metric to break the tie, where the simplicity metric is defined here as $1/A$ (A = the total number of assumptions made in an explanation). Our empirical results to be presented later in this chapter confirm that coherence is indeed a better measure in the plan recognition domain.

3.2.3 Computing the Coherence Metric

The coherence metric as defined above can be efficiently computed. We assume that the proof graph contains no cycles, since circular justification is not considered a good trait of an explanation. Using a standard depth-first graph search algorithm [Aho *et al.*, 1974], it can be readily shown that C can be computed in time

$$O(l^2 \cdot N + l \cdot e)$$

where

l = the total number of input atoms,

N = the total number of nodes in the proof graph, and

e = the total number of directed edges in the proof graph.

The depth-first search algorithm maintains a bit vector of size l at each node in the proof graph. The i-th position of the bit vector at each node is to record if there is a directed path from this node to the input atom n_i. After each child node and its descendant nodes are visited in a depth-first manner, a bit-or operation is taken on the bit vector of the parent node and that of the child node in order to update the parent node's bit vector. After traversing in depth-first order all the nodes in the proof graph so that every bit vector at each node is updated, the algorithm will consider each pair of input atoms and look at the corresponding positions in the bit vector at every node to see if there is some node that connects each pair of input atoms. The number of such pairs of connected input atoms is the sum $\sum_{1 \leq i < j \leq l} N_{i,j}$.

3.3 Finding Coherent Explanations

As mentioned in Chapter 2, finding the simplest abductive explanation has been shown to be NP-hard [Reggia *et al.*, 1983; Reggia *et al.*, 1985; Allemang *et al.*, 1987; Bylander *et al.*, 1989]. Unfortunately, finding the most coherent explanation is also NP-hard. We present a proof that a specialized instance of our problem of finding the most coherent explanation is NP-hard. The specialized optimization problem is: finding the maximally coherent explanation that satisfies simple contradiction restrictions in a two-level, propositional abduction model.

We will denote the coherence value of an explanation E as $\mathcal{C}(E)$. We show that the corresponding decision problem is NP-complete.

3.3.1 An NP-Completeness Proof

Definition 3.2 MAXIMALLY COHERENT EXPLANATION (MCE)

INSTANCE : A set O of observations, a set A of assumptions, a relation $M \subseteq A \times O$ (where $\langle a, o\rangle \in M$ denotes $a \rightarrow o$), a collection C of subsets of A (where each subset of A in C is taken to mean that the conjunction of the assumptions in the subset is contradictory), and a positive real number $K < 1$. Define an explanation E to be a graph $\langle O \cup A', M'\rangle$ with nodes $O \cup A'$ and edges M' such that $A' \subseteq A$, $M' \subseteq M$, $\{a | \langle a, o\rangle \in M'\} = A'$, and for every $C_i \in C$, $C_i \not\subseteq A'$. (The last condition ensures that A' is consistent.)

QUESTION : Is there an explanation $E = \langle O \cup A', M'\rangle$ such that $\mathcal{C}(E) \geq K$?

Theorem 3.1 *The* MCE *problem is NP-complete.*

Proof

It is clear that MCE is in NP. A nondeterministic algorithm for it need only guess some graph $\langle O \cup A', M'\rangle$ and check to see whether the graph constitutes an explanation and whether $\mathcal{C}(E) \geq K$. This can be easily done in (nondeterministic) polynomial time.

To satisfy the second requirement of NP-completeness, we will reduce the known NP-complete problem HITTING SET [Garey and Johnson, 1979] to MCE. The HITTING SET problem is :

INSTANCE : A collection D of subsets of a set S, and a positive integer L.

QUESTION : Does S contain a *hitting set* for D of size L or less, that is, a subset $S' \subseteq S$ with $|S'| \leq L$ and such that S' contains at least one element from each subset in D?

Given an instance of the HITTING SET problem $\langle S, D, L\rangle$, where $|S| = n$, we construct an instance of the MCE problem as follows:

$$O = \{o_1, o_2, \ldots, o_{2n-1}, o_{2n}\}$$
$$A = S = \{a_1, a_2, \ldots, a_n\}$$
$$M = \{\langle a_1, o_1\rangle, \langle a_1, o_2\rangle, \langle a_2, o_3\rangle, \langle a_2, o_4\rangle, \ldots, \langle a_n, o_{2n-1}\rangle, \langle a_n, o_{2n}\rangle\}$$
$$C = D$$
$$K = \frac{n-L}{2n(2n-1)/2}$$

Clearly, the construction of such an instance takes deterministic polynomial time. It remains to prove that D has a hitting set H of size L or less if and only if there is an explanation $E = \langle O \cup A', M'\rangle$ such that $\mathcal{C}(E) \geq K$.

($\Rightarrow$) Suppose D has a hitting set H where $|H| \leq L$. Let $E = \langle O \cup A', M' \rangle$ where $A' = S - H = \{a_{i_1}, a_{i_2}, \ldots, a_{i_{n-|H|}}\}$, $M' = \{\langle a_{i_1}, o_{2i_1-1} \rangle, \langle a_{i_1}, o_{2i_1} \rangle, \langle a_{i_2}, o_{2i_2-1} \rangle, \langle a_{i_2}, o_{2i_2} \rangle, \ldots\}$. That is, there are two edges connecting every assumption in A' to its two corresponding observations in the explanation. Then

$$\mathcal{C}(E) = \frac{n - |H|}{2n(2n-1)/2}$$

Since in addition $K = \frac{n-L}{2n(2n-1)/2}$ and $|H| \leq L$, it follows that $\mathcal{C}(E) \geq K$.

To prove that for each $C_i \in C$, $C_i \not\subseteq A'$, assume otherwise. Then for some $C_i \in C$, $C_i \subseteq A'$. Since $H \cap A' = \emptyset$, this implies $H \cap C_i = \emptyset$, contradicting the fact that H is a hitting set for D $(= C)$.

($\Leftarrow$) Suppose there is an explanation $E = \langle O \cup A', M' \rangle$ such that $\mathcal{C}(E) \geq K$.

$$\mathcal{C}(E) = \frac{|A'|}{2n(2n-1)/2}$$

Since $\mathcal{C}(E) \geq K = \frac{n-L}{2n(2n-1)/2}$, it follows that $|A'| \geq n - L$.

Let $H = S - A'$. Then $|H| = n - |A'| \leq L$. To prove that H is a hitting set for D, assume otherwise. That is, there is a subset $D_i \in D$ such that $D_i \cap H = \emptyset$. Since in addition $A' = S - H$, and $D_i \subseteq S$, it follows that $D_i \subseteq A'$. Since $D_i \in C$, this implies that the set of assumptions A' is inconsistent, which contradicts that fact that E is an explanation. □

3.3.2 Heuristic Search

Since our abduction problem of finding the most coherent explanation contains as a special case the abduction model formalized in the proof, our problem is clearly computationally intractable. This justifies the use of heuristic beam search in the AAA algorithm to compute the (approximately) best explanations, where the best explanations are those with the highest coherence metric, and ties are broken based on the simplicity metric of $1/A$ (A is the number of assumptions made). Note that our use of beam search to efficiently find the best explanations in the plan recognition domain is in contrast to the work of [Charniak, 1986; Charniak and Goldman, 1991] which used marker passing to restrict the search for explanations.

3.4 The AAA Algorithm in Plan Recognition

We now describe some aspects of the AAA algorithm that are specific to the plan recognition domain.

3.4.1 Computing Cross Products

As mentioned in Chapter 2 where the AAA algorithm is presented, computing the cross product of two labels (i.e., combining some environment of a label with some environment of a second label) is an expensive operation. Hence, when generating the combined environments in the cross product of two labels, the algorithm tries to generate the promising combined environments first. As soon as the number of combined environments generated exceeds β_{intra}, AAA returns these combined environments as the cross product of the two labels. This is justified since only the best β_{intra} environments are retained anyway. Considerable computational efforts can be saved in this way by not computing the entire cross product.

In order to quickly identify and compute the promising combined environments first, AAA estimates the goodness of a combined environment $E_i \cup E_j$ by the number of action types (like shopping, robbing, etc.) that are common to both environments E_i and E_j. The higher the number of shared action types, the better the estimated quality of the combined environment, since shared action types usually result in more unified assumptions and hence a higher coherence. Ties are broken based on $1/A$, where A = the sum of the number of assumptions in E_i and E_j.

We use this estimate instead of the actual coherence metric value of a combined environment since this estimate can be more quickly computed than the coherence metric, and hence serves the useful role of rapidly focusing the algorithm on generating the most promising environments first. Once the algorithm decides to combine two environments E_i and E_j, $E_i \cup E_j$ is generated and its coherence metric value is computed.

3.4.2 Consistency Checking

In the plan recognition domain, consistency checking is accomplished in the following ways:

1. There is a list of *assumption-nogoods*, where each assumption-nogood consists of a list of assumptions E such that an explanation whose assumptions include $E\sigma$ (some instance of the assumptions in an assumption-nogood) is unacceptable. These assumption-nogoods are used to ensure that appropriate role-filler assumptions are made when some high-level plan/action is assumed. For instance, $(go\text{-}step(S,G), goer(G,P))$ is an assumption-nogood, where $go\text{-}step(S,G)$ denotes "the go-step of S is G" and $goer(G,P)$ denotes "the goer of G (i.e., the agent of the go-action G) is P". This assumption-nogood ensures that when an explanation assumes that a go-action G is a substep of some high-level plan S, it is not allowed to simply assume the role-filler $goer(G,P)$, but must also assume that some appropriate role-filler exists between S and P, such as $shopper(S,P)$ ("the shopper of S is P"), or $robber(S,P)$ ("the robber of S is P"), etc. Strictly speaking, an explanation that violates an assumption-nogood is not inconsistent; it is simply not a preferred explanation since inappropriate role-filler assumptions are made. For convenience, we have chosen to enforce this explanation preference as part of consistency checking.

2. Other consistency checks are achieved via procedural code, including:

   - There is a list of binary predicates, known as *unique-predicates*, which is used by the consistency checking procedure to ensure that any two atoms that have the same unique-predicate and the same first argument must also have the same second argument. Examples of these unique-predicates include $source\text{-}go(G,S)$ (the source of a go-action G is S), $dest\text{-}go(G,D)$

(the destination of a go-action G is D), etc. That is, consistency checking enforces rules such as

$source\text{-}go(G, S1) \wedge source\text{-}go(G, S2) \rightarrow S1 = S2$

which states that the source of a go-action must be unique.

- The sort (type) of an object introduces an important constraint in the plan recognition domain. The knowledge base for this domain includes a subsort-supersort hierarchy that explicitly gives the various sort relationship, like a gun is a weapon, a bus is a vehicle, etc. During consistency checking, if some object is discovered to be of two non-compatible sorts, then a contradiction is detected. Two sorts $s1$ and $s2$ are non-compatible if an object cannot be of both sorts simultaneously. Non-compatibility of sorts is inferred from the subsort-supersort hierarchy in the knowledge base. Also, during consistency checking, if two assumptions $inst(X, s1)$ ("X is an instance of objects of sort $s1$") and $inst(X, s2)$ are found in an environment such that $s1$ is a subsort of $s2$, then the rule $inst(X, s1) \rightarrow inst(X, s2)$ will be added to the explanation and $inst(X, s2)$ will be removed from the environment (since it is subsumed by $inst(X, s1)$).

  In addition, there are rules in the knowledge base of the form $inst(X, A) \wedge role\text{-}filler(X, Y) \rightarrow inst(Y, B)$ that constrains the sorts of objects. An example of such a rule is $inst(R, robbing) \wedge weapon\text{-}rob(R, W) \rightarrow inst(W, weapon)$. This rule states that the weapon W used in a robbing plan R ($weapon\text{-}rob(R, W)$) is of type weapon. Consistency checking enforces such rules, and it also adds the rule to an explanation and removes the subsumed consequent $inst(W, weapon)$ if it finds the three assumptions $inst(R, robbing)$, $weapon\text{-}rob(R, W)$, and $inst(W, weapon)$ in an environment.

  Note that checking and adding rules like $inst(X, s1) \rightarrow inst(X, s2)$ and $inst(X, A) \wedge role\text{-}filler(X, Y) \rightarrow inst(Y, B)$ represent a limited form of forward-chaining during consistency checking.

- Consistency checking also enforces temporal constraints, including rules such as:

  $plan\text{-}step(X, X) \rightarrow false$

  $plan\text{-}step(X, Y) \wedge precede(X, Y) \rightarrow false$

  $inst(X, plan) \wedge step\text{-}1(X, S1) \wedge step\text{-}2(X, S2) \rightarrow precede(S1, S2)$

  $plan\text{-}step(X, Y) \wedge precede(X, Z) \rightarrow precede(Y, Z)$

  $plan\text{-}step(X, Y)$ denotes "Y is a substep of plan X", $precede(X, Y)$ denotes "X precedes Y", $step\text{-}1(X, S1)$ denotes "$S1$ is the first substep of plan X", and $step\text{-}2(X, S2)$ denotes "$S2$ is the second substep of plan X".

3.5 An Illustrative Example

Consider the following example which is one of the 50 narrative texts that ACCEL was tested on:

"Bill went to the liquor-store.

He pointed a gun at the owner."

The sentences translate into the following conjunction of atoms:

$inst(go6, going) \wedge goer(go6, bill6) \wedge name(bill6, bill) \wedge$

$\textit{dest-go}(go6, ls6) \wedge \textit{inst}(ls6, \textit{liquor-store}) \wedge \textit{precede}(go6, point6) \wedge$
$\textit{inst}(point6, \textit{pointing}) \wedge \textit{agent-point}(point6, bill6) \wedge \textit{instr-point}(point6, gun6) \wedge$
$\textit{inst}(gun6, gun) \wedge \textit{patient-point}(point6, o6) \wedge \textit{own}(o6, ls6)$

The knowledge base axioms relevant to this example are listed below:

$$\begin{array}{rcl}
\textit{inst}(G, \textit{going}) & \leftarrow & \textit{inst}(S, \textit{shopping}) \wedge \textit{go-step}(S, G) \\
\textit{goer}(G, P) & \leftarrow & \textit{inst}(S, \textit{shopping}) \wedge \textit{go-step}(S, G) \wedge \\
 & & \textit{shopper}(S, P) \\
\textit{dest-go}(G, P) & \leftarrow & \textit{inst}(S, \textit{shopping}) \wedge \textit{go-step}(S, G) \wedge \\
 & & \textit{store}(S, P) \\
\textit{inst}(P, \textit{liquor-store}) & \leftarrow & \textit{inst}(S, \textit{liqst-shopping}) \wedge \textit{store}(S, P) \\
\textit{inst}(G, \textit{going}) & \leftarrow & \textit{inst}(R, \textit{robbing}) \wedge \textit{go-step}(R, G) \\
\textit{goer}(G, A) & \leftarrow & \textit{inst}(R, \textit{robbing}) \wedge \textit{go-step}(R, G) \wedge \\
 & & \textit{robber}(R, A) \\
\textit{dest-go}(G, P) & \leftarrow & \textit{inst}(R, \textit{robbing}) \wedge \textit{go-step}(R, G) \wedge \\
 & & \textit{place-rob}(R, P) \\
\textit{inst}(P, \textit{pointing}) & \leftarrow & \textit{inst}(R, \textit{robbing}) \wedge \textit{point-weapon-step}(R, P) \\
\textit{agent-point}(P, A) & \leftarrow & \textit{inst}(R, \textit{robbing}) \wedge \textit{point-weapon-step}(R, P) \wedge \\
 & & \textit{robber}(R, A) \\
\textit{instr-point}(P, W) & \leftarrow & \textit{inst}(R, \textit{robbing}) \wedge \textit{point-weapon-step}(R, P) \wedge \\
 & & \textit{weapon-rob}(R, W) \\
\textit{patient-point}(P, A) & \leftarrow & \textit{inst}(R, \textit{robbing}) \wedge \textit{point-weapon-step}(R, P) \wedge \\
 & & \textit{victim-rob}(R, A)
\end{array}$$

The first axiom asserts that if S is a shopping event and the go-step of S is G, then G is a going event; the second axiom asserts that if S is a shopping event, the go-step of S is G, and the shopper of S is P, then the goer of G (i.e., the agent of the going event G) is P; and so on. These axioms are formulated in such a way that higher-level plans (like shopping and robbing) together with the appropriate role-filler assumptions (like $\textit{shopper}(S, P)$, $\textit{robber}(R, A)$) imply the input atoms representing the observed actions (like going to a store and pointing a gun).

After the first sentence is processed, the best interpretation ACCEL computed (as determined by the coherence metric) is shown in Figure 3.2. The interpretation is that Bill was shopping in the liquor-store. After processing the second sentence, the best interpretation changes to one where Bill was robbing the liquor-store (Figure 3.3). The total time ACCEL took to process these two sentences is 0.83 minutes on a Sun Sparc 2 workstation.

Note that ACCEL is able to pick the shopping interpretation over the robbing interpretation (Figure 3.4) after processing sentence 1 since liquor-store shopping explains why Bill went to the liquor-store, but robbing does not explain why Bill went to the liquor-store (as opposed to say a supermarket or a bank). Since the coherence metric favors explanations that connect together more input atoms, the shopping interpretation is preferred.

For comparison, we also ran ACCEL using simplicity as the criterion for selecting the best explanation. In this case, ACCEL is indifferent between the robbing

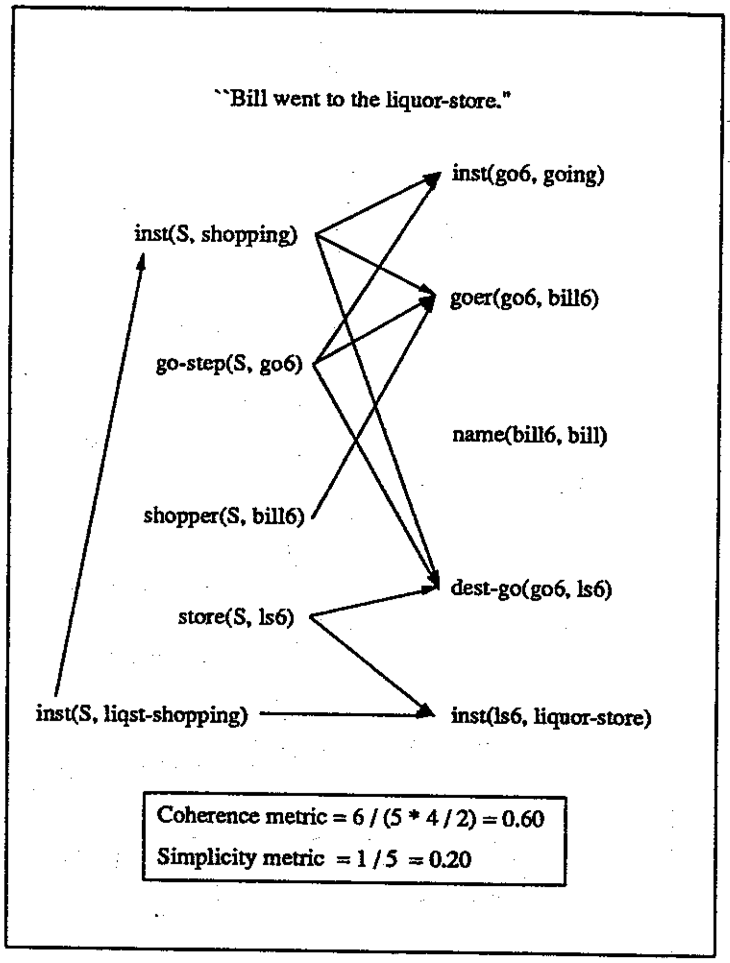

Figure 3.2: The shopping interpretation after the first sentence.

interpretation (Figure 3.3) and the shopping interpretation (Figure 3.5) after processing the two sentences. Both interpretations have the same number of assumptions (12). Note that even if we change the definition of simplicity so that input atoms do not count as assumptions, simplicity will still pick the wrong interpretation since the shopping interpretation will then have fewer assumptions (4) than the robbing interpretation (7). Charniak's metric of $E - A$ does not help either, since for the shopping interpretation, $E - A = 4 - 4 = 0$, and for the robbing interpretation, $E - A = 7 - 7 = 0$.

3.6 Empirical Results

Evaluating natural language processing systems is becoming an increasingly important issue. For instance, there is ongoing work to evaluate NLP systems that perform information extraction from unconstrained texts [Lehnert and Sundheim, 1991]. We have completed an evaluation of ACCEL on a test suite of 25

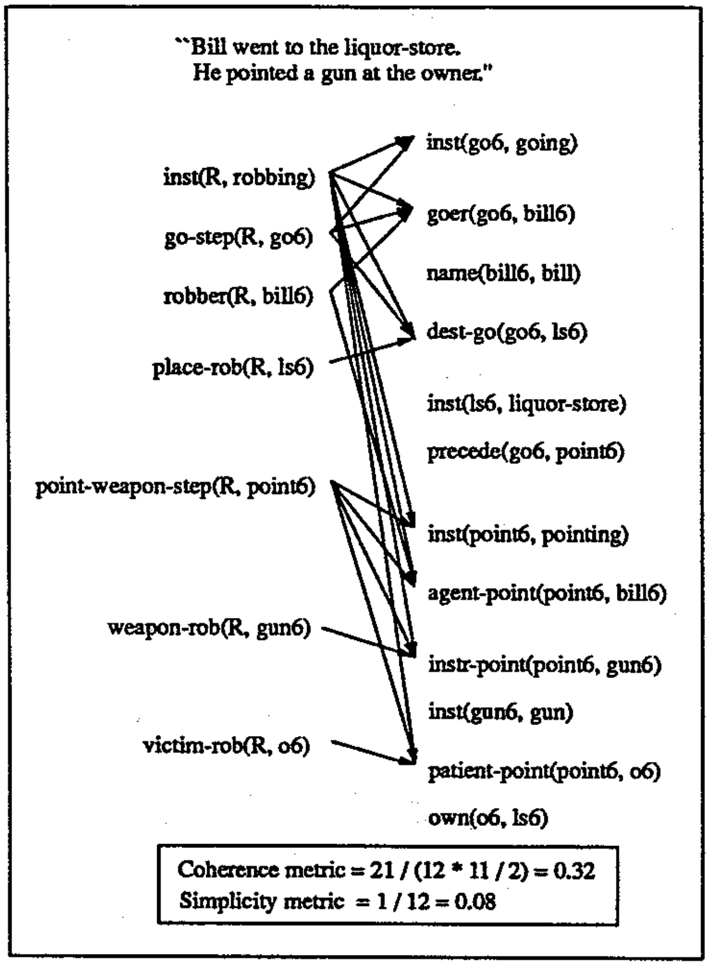

Figure 3.3: The robbing interpretation after the second sentence.

examples taken from Robert Goldman's PhD thesis [Goldman, 1990]. We chose this set of examples to test our system since we are aware of no other pre-existing set of test data for plan recognition, and it also facilitates comparison between different approaches.

The knowledge base was initially constructed so as to handle this set of 25 examples. In order to test for generality, Ray Mooney came up with another 25 test examples unbeknown to the knowledge base builder (the author). The intent is that these additional examples will test for other novel combinations and sequences of actions that the knowledge base constructed for the initial 25 examples in principle should be able to handle. We will call the first set of 25 examples the *training* examples, and the second set of 25 examples the *test* examples. Note that our evaluation methodology is similar to that of Goldman, except that we tested ACCEL on a *different* set of 25 test examples whereas Goldman tested his system's ability to pair up an additional set of 25 *similar* examples to the initial 25 examples. Hence, our evaluation criterion is tougher.

The plans in the knowledge base include shopping, robbing, restaurant

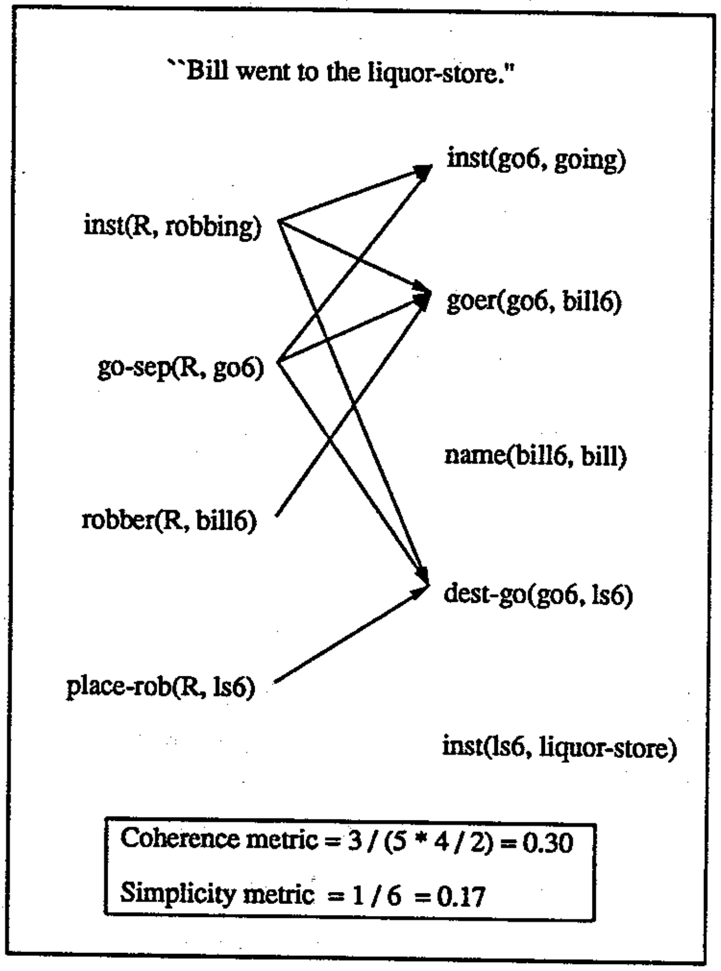

Figure 3.4: The robbing interpretation after the first sentence.

dining, traveling in a vehicle (bus, taxi, or plane), partying, and jogging. Each of these plans in turn has subplans, and some of the plans contain recursive subplans. For instance, traveling by plane includes the subplan of traveling (in some vehicle) to the airport to catch a plane. For each example, a set of input atoms representing the sentences is given to ACCEL. To give a sense of the size of our examples and the knowledge base used, there is a total of 107 KB rules, 45 assumption-nogoods, and 70 taxonomy-sort symbols. Every taxonomy-sort symbol p will add an axiom (in addition to the 107 KB rules) of the form $inst(X,p) \rightarrow inst(X, supersort\text{-}of\text{-}p)$. The average number and maximum number of input atoms per example are 12.6 and 26 respectively. The knowledge base and the 50 examples are included in Appendix A.

For each example, the correct explanation was determined based on the author's intuition before running the example. To measure the quality of an explanation computed by ACCEL, we compared it to the correct explanation and recorded three error rates: the recall error rate R = the number of missing assumptions divided by the number of assumptions in the correct explanation, the precision error

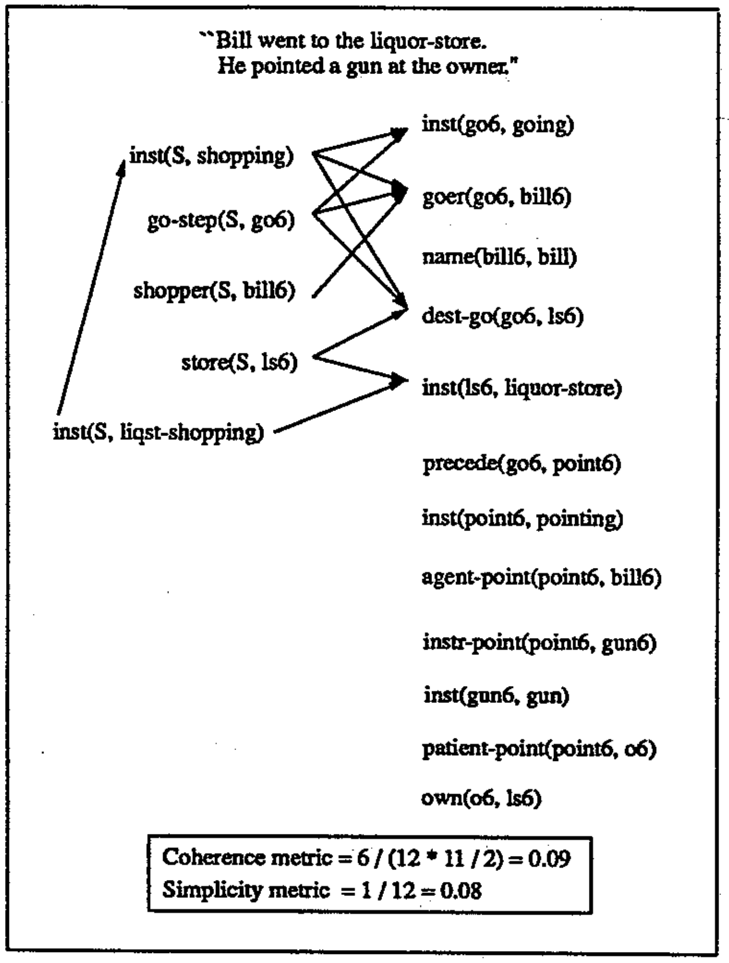

Figure 3.5: The shopping interpretation after the second sentence.

rate P = the number of excess assumptions divided by the number of assumptions in the computed explanation, and the overall error rate O = the average of the recall and precision error rates. (We used similar quality measures and terminology as in [Lehnert and Sundheim, 1991].) For example, a computed explanation that matches exactly the correct explanation has $R = P = O = 0\%$, and a computed explanation with abductive assumptions that are completely disjoint from those of the correct explanation has $R = P = O = 100\%$. If more than one best explanations are computed for an example, we take the error rates for the example to be the average of the error rates over all the best explanations.

We set the beam widths of AAA at $\beta_{inter} = 10$ and $\beta_{intra} = 30$ when processing the set of 50 examples. We ran ACCEL on all the examples (training and test) using two different evaluation metrics: the coherence metric (breaking ties based on simplicity) and the simplicity metric. The empirical results are summarized in Table 3.1, which shows the average recall (R), precision (P), and overall (O) error rates for the training examples, test examples, and all examples. The average run time per example is 1.83 minutes on a Sun Sparc 2 workstation.

Example type	Coherence			Simplicity		
	R	P	O	R	P	O
Training	0.2%	0%	0.1%	26%	25%	25%
Test	2%	2%	2%	39%	38%	38%
All	1.1%	1%	1%	32%	31%	32%

Table 3.1: Empirical results comparing coherence and simplicity.

As demonstrated by our empirical results, coherence consistently performs better than simplicity on the set of 50 examples. The empirical evidence supports our claim that coherence is a better measure than simplicity for determining the quality of an explanation for a narrative text.

In a few of the examples (training example #17, 23, 24, test examples #21, 22), there are several equally coherent interpretations computed by ACCEL. For instance, on training example #24, "Bill took a taxi", ACCEL found that there are four best, equally coherent interpretations, namely, Bill went shopping, robbing, restaurant dining, or took the taxi to a place so as to board some other vehicle for further traveling. Similar situations occurred in each of the other four examples mentioned above. In all these cases of equally best coherent interpretations, the human reader's best interpretation is not to abduce any high-level plans (i.e., infer nothing other than Bill took a taxi). In such situations, a reasonable strategy that a system like ACCEL could adopt is not to settle on any one of the interpretations if there are multiple best interpretations. As such, we assign a perfect score ($R = P = O = 0\%$) for these examples in which there are multiple most coherent interpretations and the best interpretation is just assuming the input atoms. (If we adopt a more stringent criterion and take the error rates for these five examples to be the average of the error rates over all their best explanations, then the overall error rate for all the 50 examples increases to 6.5%.)

The correct best explanations are selected for all but one of the 25 training examples. The one training example that ACCEL did not score perfectly is example #21. The sentences of this example are: "Bill took a bus to a restaurant. He drank a milkshake. He pointed a gun at the owner. He got some money from him." ACCEL correctly inferred that Bill was dining at the restaurant as well as robbing it. It correctly abduced 24 of the 25 assumptions of the correct abductive explanation, missing only one assumption, $go\text{-}step(D, go21)$ (i.e., that going to the restaurant is also a substep of dining at the restaurant, in addition to being a substep of robbing). Note that Goldman's system made a similar error of not connecting the going action to both dining and robbing. ACCEL committed this mistake because it assumed that an action cannot be a part of two high-level plans, which is not true in general. In addition to allowing an action to be part of two plans, ACCEL must also consider non-minimal explanations in order to process this example correctly. This is because both assumptions $go\text{-}step(D, go21)$ and $go\text{-}step(R, go21)$ are needed in the preferred explanation but either one of them is sufficient to explain inst(go21,going) (i.e., Bill's going to the restaurant).

On the test set of 25 examples, ACCEL correctly processed 22 of them using the coherence metric. On test example #19 "Mary got a gun. She went to the supermarket. She found the milk on the shelf. She got some money from the cashier." ACCEL made a similar mistake of not abducing that the go-action is a substep of shopping in addition to a substep of robbing.

On test example #8 "Fred got a gun. He went to the restaurant. He packed a suitcase.", ACCEL could not arrive at the interpretation that Fred was packing a suitcase so as to escape and get away with his robbery, since the knowledge base did not have an escape substep in its rob plan. Furthermore, ACCEL tried to connect the input sentences "overzealously". In addition to inferring that Fred robbed the restaurant, ACCEL also inferred that Fred packed a suitcase so as to catch a plane to go to and dine at the same restaurant! ACCEL chose this interpretation since the interpretation connected the packing of suitcase in the third sentence to the restaurant mentioned in the second sentence.

On test example #11 "Jane took a taxi to the airport. She pointed a gun at the taxi-driver.", ACCEL erroneously inferred that the taxi-trip was the going-step of the robbing event and that the airport is the place of robbery, although it did correctly infer that the victim of the robbery is the taxi-driver and the gun-pointing action is part of the robbing event. These problems indicate that more common sense knowledge about plans is needed to achieve effective plan recognition.

3.7 Comparison with Other Approaches

Several research efforts have focused on abduction as an important inference process in natural language understanding [Hobbs *et al.*, 1988] and plan recognition [Charniak, 1986]. In the area of natural language understanding, [Hobbs *et al.*, 1988] describes the use of abduction in solving the four local pragmatics problems of text understanding. This work differs from ours in that unlike their emphasis on linguistic issues like reference resolution and syntactic ambiguity resolution, ACCEL is concerned with recognizing high level plans from the actions given in a text.

The work of [Charniak, 1986] and [Charniak and Goldman, 1989; Charniak and Goldman, 1991] are most similar to ours. However, explanations are evaluated based on explanatory coherence in our work, as opposed to the simplicity criterion of [Charniak, 1986] and the probability criterion of [Charniak and Goldman, 1989; Charniak and Goldman, 1991]. The empirical results reported in the last section support our claim that the coherence metric is superior to the simplicity metric.

Charniak and Goldman [Charniak and Goldman, 1989; Charniak and Goldman, 1991] have adopted the Bayesian probabilistic approach to plan recognition and text understanding. In this approach, an explanation is selected based on the conditional probabilities of the abduced events given the observations stated in the input text. As mentioned earlier, the first 25 training examples used in our system were taken from Goldman's thesis and these examples have been successfully processed based on finding the most probable explanation.

As reported in the last section, ACCEL achieved results similar to Goldman's system on the 25 training examples. Note that almost all the best explanations are found even though our knowledge base does not contain any probabilistic or likelihood information.

This may seem surprising. In the probabilistic approach, the primary purpose of *a priori* probabilities is to select a most likely explanation when there are otherwise multiple competing explanations. For instance, in the sentence "John went to the supermarket.", higher a priori probability of someone shopping at the supermarket as compared to robbing the supermarket enables the supermarket shopping interpretation to be selected over the supermarket robbing interpretation. In our system, we achieve an analogous effect by having an axiom in the knowledge base

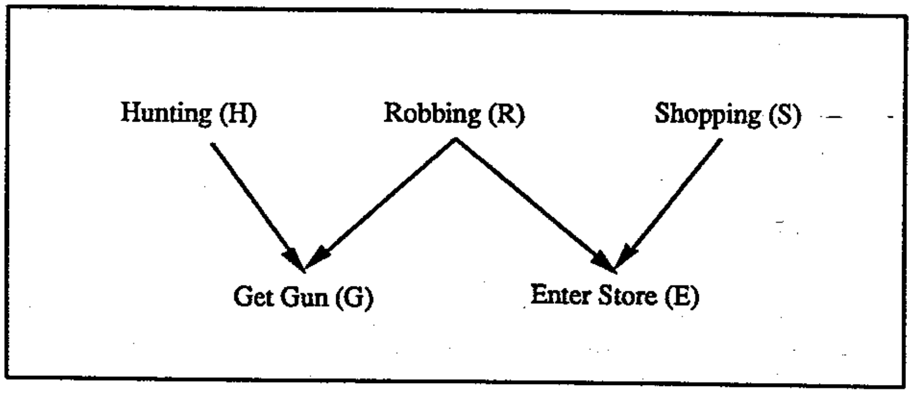

Figure 3.6: The Bayesian network for "John got a gun. He entered the grocery store."

that explains supermarket in terms of supermarket shopping, but the knowledge base does *not* have the corresponding axiom for "supermarket robbing" that explains supermarket. That is, we have the following axiom in the knowledge base:

$$inst(S, smarket\text{-}shopping) \wedge store(S, P) \rightarrow inst(P, smarket)$$

but *not*

$$inst(S, smarket\text{-}robbing) \wedge store(S, P) \rightarrow inst(P, smarket).$$

This is justified since supermarket shopping is a commonly occurring plan, but not supermarket robbing. (In fact, the knowledge base does not have the high-level plan supermarket-robbing. Only robbing is present as a high-level plan in the knowledge base.) It then follows that supermarket shopping is a more coherent interpretation of "John went to the supermarket." since supermarket shopping explains and thus connects both supermarket and the going action, whereas robbing only explains the going action but not supermarket.

With this style of axiomatization and the use of the coherence metric, ACCEL is able to select the correct explanation without resorting to the use of numeric probabilities. In essence, what is achieved by numeric probabilities in the probabilistic approach is accomplished by the judicious use of logical axioms. This is in contrast to the probabilistic approach which critically depends on knowledge about the numerous prior and posterior probabilities of the nodes in a Bayesian network constructed from the input sentences. In practice, such knowledge may not always be available in the required form. For example, in the probabilistic approach, in order to understand the sentences "John got a rope. He killed himself.", one needs to know the prior probability of a hanging event, the prior probability of an entity being a rope, etc, which in turn necessitates making the assumptions that there are 10^{20} things in the world, out of which there are 10^9 ropes, 10^{15} get events, 10^3 hangings, etc. Engineering an appropriate set of probabilities is a major weakness of the probabilistic approach to text understanding.

In addition, the most probable interpretation selected depends quite critically on the specific values assigned to the various probabilities, and reasonable probability values may result in the wrong interpretation being selected. For example, consider the sentences "John got a gun. He entered the grocery store." Figure 3.6 shows a simple Bayesian network constructed for these sentences.

Suppose we adopt the following reasonable estimates for the various prior and posterior probabilities:

$$P(h) = 10^{-3}; P(r) = 10^{-5}; P(s) = 10^{-2}$$

$$P(g \mid h, r) = 0.95; P(g \mid h, \overline{r}) = P(g \mid \overline{h}, r) = 0.90; P(g \mid \overline{h}, \overline{r}) = 0.01$$
$$P(e \mid r, s) = 0.95; P(e \mid r, \overline{s}) = P(e \mid \overline{r}, s) = 0.90; P(e \mid \overline{r}, \overline{s}) = 0.01$$

Note that under the above probability estimates, shopping is more probable than either hunting or robbing *a priori*. It follows that these estimates will specify a complete and consistent joint probability distribution function [Pearl, 1988]. The joint distribution corresponding to this Bayesian network is:

$$P(H, R, S, G, E) = P(H) \cdot P(R) \cdot P(S) \cdot P(G \mid H, R) \cdot P(E \mid R, S)$$

We can now compute the needed conditional probabilities by the method of [Pearl, 1988] or [Lauritzen and Spiegelhalter, 1988]. Alternatively, since this network is relatively simple, a straightforward summation of P(H, R, S, G, E) over various variable combinations yields $P(r \mid g, e) = 0.038$, $P(h \mid g, e) = 0.080$, $P(s \mid g, e) = 0.459$, and $P(h, s \mid g, e) = 0.038$. Since the conditional probability of shopping given the observations get-gun and enter-store is the highest, the chosen interpretation is that John is shopping in the store! Even though the chosen probability estimates are quite reasonable, the preferred interpretation that John is robbing the grocery store is not selected. Furthermore, suppose all the above prior and posterior probabilities remain the same except that $P(r) = 1.222 * 10^{-4}$. Then $P(r \mid g, e) \approx P(s \mid g, e) \approx 0.3249$, and for $P(r) > 1.222 * 10^{-4}$, $P(r \mid g, e)$ becomes the highest of all the conditional probabilities. This suggests that the selected interpretation is quite sensitive to slight variation in the estimated subjective probabilities. Note that the correct interpretation that John is robbing the store will be selected using our coherence metric, since the robbing interpretation has a positive coherence value compared to the zero coherence of hunting or shopping.

Besides the problem of engineering the numerous prior and posterior probabilities of the nodes in a Bayesian network, the probabilistic approach does not take into account the importance of text coherence. Selecting an interpretation based solely on the probability of propositions about the situation being described is ignoring the fact that these propositions are adjacent sentences in a natural language text, not just random facts observed in the world.

Cost-based abduction is another scheme proposed by [Hobbs *et al.*, 1988] to select an interpretation based on the cost of an abductive proof. However, as shown in [Charniak and Shimony, 1990], cost-based abduction can be given a probabilistic semantics. Therefore, cost-based abduction can be regarded as a kind of probabilistic approach, and it suffers from the same problems.

In summary, our method yields the correct interpretations without the heavy machinery of the probabilistic approach, and consistently produces more accurate interpretations than a metric based on simplicity. The approach generalized well to novel test examples, although the system sometimes constructed implausible connections between events. Our empirical results indicate that maximizing connections between observations is an important property of a good explanation in plan recognition.

Chapter 4

Diagnosis Based on Set Covering

In this chapter, we present the Generalized Set Covering (GSC) model of diagnosis [Peng and Reggia, 1990] and show how ACCEL can compute the same diagnoses as the GSC model. Furthermore, we present empirical results of ACCEL efficiently diagnosing 50 real patient test cases using a moderately sized background knowledge base. This supports our claim that a general abductive system can efficiently diagnose real problems of sizable complexity.

4.1 Generalized Set Covering

Over the past decade, Reggia and his colleagues have developed an increasingly sophisticated theory of diagnosis based on set covering and applied the theory primarily to medical disease diagnosis [Peng and Reggia, 1990].

Definition 4.1 *The basic diagnostic problem in the Generalized Set Covering (GSC) model is defined by four sets:* (D, M, C, M^+)

D*: A finite set of potential disorders*

M*: A finite set of potential manifestations (symptoms)*

$C \subseteq D \times M$*: A causation relation where* $(d, m) \in C$ *means "d may cause m"*

$M^+ \subseteq M$*: The set of observed manifestations for the current case*

$E \subseteq D$ *is called a* cover *of* M^+ *iff for each* $m \in M^+$ *there exists* $d \in E$ *such that* $(d, m) \in C$*. A cover is said to be* minimum *if its cardinality is the smallest among all covers and* irredundant *(*minimal*) if none of its proper subsets is also a cover.*

Depending on the domain, one may consider all minimum or all minimal covers of the observed symptoms as the best diagnoses.

4.2 Set-Covering-Based Diagnosis as Abduction

We can map a GSC diagnostic problem into an abduction problem in ACCEL as follows: Let the domain theory T be the set of axioms $\{d \rightarrow m | (d, m) \in C\}$, and let the input atoms $O = \bigwedge_{m \in M^+} m$. We use predicate specific abduction such that only atoms $d \in D$ are assumable.

Theorem 4.1 *The set of covers of GSC = the set of explanations in* ACCEL.

Proof Let $E = \{d_1, \ldots, d_r\}, M^+ = \{m_1, \ldots, m_s\}$.

($\Rightarrow$) Let E be a cover of M^+.

Note that $d_1 \wedge \ldots \wedge d_r \wedge T$ is consistent.

Since E is a cover of M^+, for each $m_j \in M^+$, there exists $d_i \in E$ such that $(d_i, m_j) \in C$. That is,

$$d_i \wedge T \models m_j$$

It follows that $d_1 \wedge \ldots \wedge d_r \wedge T \models m_j$ for each $m_j \in M^+$. Hence,

$$d_1 \wedge \ldots \wedge d_r \wedge T \models m_1 \wedge \ldots \wedge m_s$$

Therefore, E is an explanation for $m_1 \wedge \ldots \wedge m_s$.

($\Leftarrow$) Let E be an explanation for $m_1 \wedge \ldots \wedge m_s$. That is,

$$d_1 \wedge \ldots \wedge d_r \wedge T \models m_1 \wedge \ldots \wedge m_s$$

Since $T = \{d \rightarrow m | (d, m) \in C\}$, it follows that for each $m_j \in M^+$, there exists $d_i \in E$ such that $(d_i, m_j) \in C$. (For if there is no such $d_i \in E$, then $d_1 \wedge \ldots \wedge d_r \wedge T \wedge \neg m_j$ is consistent, contradicting $d_1 \wedge \ldots \wedge d_r \wedge T \models m_1 \wedge \ldots \wedge m_s$.)

Hence E is a cover of M^+. □

It follows from this theorem that the set of all *minimal* covers of GSC is identical to the set of all *minimal* explanations in ACCEL.

That all minimal covers of GSC are all minimal explanations in abduction also follows as a corollary of two published theorems. In [Reiter, 1987], Reiter proves (Theorem 7.1) that by appropriately axiomatizing the knowledge base, his consistency-based model can solve the GSC problem. In [Poole, 1988], Poole proves (Theorem 4.2) that, for propositional knowledge bases, proper axiomatization results in consistency-based and abductive diagnosis computing the exact same diagnoses. Since Poole uses the same representation of diagnostic knowledge to make consistency-based diagnosis model abduction as Reiter uses to make consistency-based diagnosis model GSC, it follows that abduction solves the GSC problem.

Since the logical abduction approach is based on a more expressive representation language, it can accommodate more naturally "causal chaining" [Peng and Reggia, 1990], incompatible disorders, and symptoms caused by combinations of disorders. Causal chaining can be achieved in logical abduction by allowing backward-chaining of depth greater than one. Incompatible disorders can be enforced through consistency checking by adding nogoods $d_1 \wedge d_2 \rightarrow \mathit{false}$. A symptom s that is caused by multiple, simultaneous disorders $d_1, \ldots, d_n$ can be encoded as $d_1 \wedge \ldots \wedge d_n \rightarrow s$.

A standard concern with the logical approach is that a disorder may not always cause all of its manifestations. In this case, the axiom $d \rightarrow m$ is too strong since assuming d would be inconsistent if $\neg m$ is observed. As described in [Poole, 1988], this problem is easily handled by making $d \rightarrow m$ a potential assumption rather than an axiom when the symptom is not deterministic. If assumptions are required to be atomic (as in ACCEL), then one can achieve the same effect by adding an extra unique antecedent to rules for nondeterministic symptoms, $d \wedge a \rightarrow m$, where a represents the assumption that d actually causes m in the current case.[1]

Since GSC diagnostic problems can be nicely represented as abduction problems, the remaining question is whether a general abductive system can solve such problems efficiently. As the GSC diagnostic problem is NP-hard [Reggia *et al.*, 1985], the issue then becomes whether a general abductive system can solve real problems in reasonable time and is competitive with existing set-covering algorithms. To address this issue, we tested ACCEL on the medical problem studied in [Tuhrim *et al.*, 1991], which involves determining the areas of the brain that were damaged in a stroke. We will present empirical results showing the performance of ACCEL on this domain in a later section in this chapter.

4.3 The AAA Algorithm in Set-Covering Diagnosis

Aspects of the AAA algorithm that are specific to the set-covering-based diagnosis domain include:

- No consistency checking of assumptions is performed, since in the set covering model, it is assumed that all possible disorders can consistently explain any observed symptoms.
- Factoring of assumptions is performed. Since the knowledge base axioms are propositional, factoring assumptions degenerates into collapsing identical assumptions.
- Since the knowledge base axioms are propositional, eliminating variant supersets becomes eliminating simple supersets. Such elimination of redundant supersets is performed.

4.4 An Illustrative Example

To illustrate set-covering diagnosis, consider the following example which is one of the 50 patient cases used to test ACCEL. The symptoms observed are:

- a decreased consciousness (with severity level described as drowsy),
- eye movement impairment (of type right, horizontal gaze palsy),
- facial weakness (of type right, central),
- motor weakness (of type right hemiparesis),
- decreased rapid alternating movements (of type right, mild),
- deep tendon reflex abnormality (of type right, hyperreflexia), and

[1]If minimum covers are desired, then the extra assumptions should not count as contributing to the size of the cover.

- abnormal plantar response (of type right).

In this case, there is only one minimum diagnosis: that the left frontal lobe of the brain is damaged. The knowledge base axioms used to construct the best explanation are:

$$\begin{array}{rcl}
\mathit{decloc\text{-}degree(drowsy)} & \leftarrow & \mathit{left\text{-}frontal\text{-}lobe(present)} \\
\mathit{abneom\text{-}type(hgaze\text{-}right)} & \leftarrow & \mathit{left\text{-}frontal\text{-}lobe(present)} \\
\mathit{facial\text{-}side\text{-}type(right\text{-}central)} & \leftarrow & \mathit{left\text{-}frontal\text{-}lobe(present)} \\
\mathit{weakness\text{-}type(hemiparesis\text{-}right)} & \leftarrow & \mathit{left\text{-}frontal\text{-}lobe(present)} \\
\mathit{decram\text{-}side(right\text{-}mild)} & \leftarrow & \mathit{left\text{-}frontal\text{-}lobe(present)} \\
\mathit{abndtrs\text{-}side(right\text{-}incdtr)} & \leftarrow & \mathit{left\text{-}frontal\text{-}lobe(present)} \\
\mathit{babs\text{-}side(right)} & \leftarrow & \mathit{left\text{-}frontal\text{-}lobe(present)}
\end{array}$$

Other non-minimum diagnoses require assuming that two or more regions of the brain are damaged. ACCEL computed this minimum diagnosis in 1.8 seconds and this minimum diagnosis also agrees with the diagnosis of a human physician.

4.5 Empirical Results

We tested ACCEL on the medical problem studied in [Tuhrim *et al.*, 1991], which specifies 25 brain areas (e.g. right frontal lobe) whose damage can explain 37 basic symptom types (e.g. impaired gag reflex). The knowledge base is quite large, consisting of 648 rules of the form: $d \rightarrow m$. We were only able to obtain 50 of the original 100 cases from the authors of the initial study, each consisting of an average of 8.56 symptoms.

ACCEL efficiently computed all of the minimal explanations in an average of 2.4 seconds per case on a Sun Sparc 2 workstation. Unfortunately, we could not compare this result to that obtained in the original study, since no information on run time was provided. However, it is clear that a general abductive system can solve real diagnostic problems in reasonable time.

The original study compared the diagnosis of an expert physician to that obtained with set covering using several explanation-evaluation criteria: minimum, minimal, most-probable, and minimum-collapsed covers. The most-probable explanation is computed using methods described in [Peng and Reggia, 1990]. The minimum-collapsed criterion is domain-specific and based on the spatial layout of the brain.

Since abduction computes the same explanations as set covering when given the same evaluation criteria, ACCEL should replicate the accuracy results of the original study. As discussed in the original study, minimality is too unrestrictive to produce useful results (ACCEL returned an average of 26.6 minimal diagnoses per case). With minimum cardinality, ACCEL produced an average of only 4.6 diagnoses per case. In 44% of the cases, one of these diagnoses matches the expert's exactly; and in another 46% of the cases, one of the system's diagnoses was a subset or superset of the expert's (called a "close match" in [Tuhrim *et al.*, 1991]). The remaining 10% of the cases have a diagnosis that either partially matches the expert's (2%) or all of the diagnoses are totally wrong (8%). These results are slightly better than those reported in the original study: 6.5 diagnoses/case with 40% exact, 38% close, 5%

partial, 17% wrong. This is presumably due to the fact that our results are based on only 50 of the original 100 cases. The most-probable and minimum-collapsed metrics reported in the original study performed even better. In [Tuhrim *et al.*, 1991], it is claimed that, although there have been no direct comparisons, the results from any of the covering metrics appear more promising than those obtained from standard rule-based approaches to this problem.

An alternative diagnostic accuracy measure is as follows. For each case, we count the number of disorders (n) that ACCEL correctly identifies as being either present or absent from the case. Let N be the total number of possible disorders. ($N = 25$ in the current study since there are 25 possible brain areas.) n/N gives the proportion of the possible disorders whose presence or absence are correctly predicted by ACCEL. (If there are multiple minimum diagnoses for a case, we take the average over all minimum diagnoses.) Using this measure, ACCEL achieved an average of 91% accuracy over all 50 patient cases.

In summary, the results presented in this chapter demonstrate that our general-purpose logic-based abductive system can effectively represent and efficiently solve large realistic problems suitable for set-covering methods. Consequently, the desirability of the existing special-purpose approach for such problems is lessened. The logical approach is more general and flexible, yet capable of efficiently solving problems in this more restrictive class.

Chapter 5

Model-Based Diagnosis

5.1 Introduction

Model-based diagnosis has recently been an active area of research in artificial intelligence. In model-based diagnosis, an underlying model of a device's correct structure and behavior is given to a diagnostic system. Diagnosis proceeds from first principles using such a model of a device to explain discrepancies between its faulty and normal behavior. This model-based approach has some advantages over the associational, heuristic rule-based approach of conventional expert systems. The model-based approach is compositional in that it lets us define models for a library of basic components, and it works on all systems composed from those components. The system designer can focus on getting the component models right, leading to more robust and sound diagnostic systems. The potential is also better for verification of the underlying knowledge base.

However, much of the research in model-based diagnosis has taken the *consistency-based* approach and has been applied primarily to devices with static, persistent states such as combinational logic circuits [Davis, 1984; de Kleer and Williams, 1987; Reiter, 1987; de Kleer and Williams, 1989]. In the consistency-based approach, a diagnosis is a set of normality and abnormality assumptions about the device components that are *consistent* with the observations and the system description. This is in contrast with an *abductive* approach to diagnosis, in which normality and abnormality assumptions about the device components together with the system description must *imply* or *explain* the observations. The work of [de Kleer and Williams, 1987; Reiter, 1987; de Kleer and Williams, 1989] applied consistency-based diagnosis to logic circuits, while the work of [Ng, 1991; Ng, 1990] dealt with consistency-based diagnosis for continuous dynamic systems.

David Poole has convincingly argued that both consistency-based and abductive approaches appear able to solve the same range of diagnostic problems given the appropriate axiomatization of domain knowledge [Poole, 1988; Poole, 1989b]. Specifically, he showed that the two approaches are equivalent for propositional theories. Konolige extended the conditions under which equivalence holds to general first-order causal theories allowing for correlations, uncertainty, and acyclicity in the causal structure [Konolige, 1992]. In view of such formal equivalence results, issues such as ease of representation and computational efficiency are most important. In this chapter, we present empirical results suggesting that a number of diagnostic problems, ranging from combinational logic circuits to continuous dynamic systems such as a proportional temperature controller and the water balance system of the human kidney, can be effectively represented and efficiently diagnosed using an abductive approach.

Research in model-based diagnosis can also be classified according to whether information about fault models is utilized in diagnosis. The *normality-based* approach of [Reiter, 1987; de Kleer and Williams, 1987] does not utilize fault models and any misbehavior differing from the correct functioning of a device can be diagnosed. However, the lack of fault models may result in hypothesizing implausible faults. On the other hand, the work of [Dvorak and Kuipers, 1989] is *fault-based* in that the fault models are *a priori* determined and given to the diagnostic system. Hence, unanticipated faults are not detected. Finally, the diagnostic systems Sherlock [de Kleer and Williams, 1989], GDE+ [Struss and Dressler, 1989] as well as ACCEL combine both normality-based and fault-based diagnosis in that information about fault models is used in diagnosis and any deviation from the correct behavior can be diagnosed.

5.2 The AAA Algorithm in Model-Based Diagnosis

We now describe some aspects of the AAA algorithm that are specific to the model-based diagnosis domain.

- We use predicate specific abduction. The assumable atoms include component behavioral mode assumptions, which are of three types: (1) a component is normal; (2) a component is in some known fault mode; or (3) a component is abnormal but not in any known fault mode. Other assumable atoms are "auxiliary" assumptions which include assumptions that the input values of a device are as given, and in dynamic system diagnosis, that some qualitative magnitude is positive/negative, that two qualitative values obey the corresponding value constraint, etc. (More details about these auxiliary assumptions will be provided in later sections when we give the knowledge base axioms of the various devices.)

- Both factoring of assumptions and removing variant supersets are not performed. Each explanation of the input atoms consists of different combinations of component behavioral mode assumptions (normality, fault mode, and abnormality) and so factoring and removing supersets are not needed.

- Explanations are evaluated based on simplicity, where the best explanation is one with the least number of components that are not normal, which include components that are in some known fault mode and those that are not. Normality assumptions and auxiliary assumptions are "free" and do not affect the simplicity metric of an explanation. If two explanations have the same number of components that are not normal, then the one with the most number of components that are in some known fault mode is preferred.

- As in the plan recognition domain, when generating the combined environments in the cross product of two labels, the algorithm tries to generate the promising combined environments first. In this domain, the simplicity of the combined environment $E_1 \cup E_2$ of two environments E_1 and E_2 can be precisely computed without first generating $E_1 \cup E_2$, by summing the number of components that are not normal in both E_1 and E_2. At the time of generating the combined environment, the algorithm also performs consistency checking to ensure that a component cannot be in two different behavioral modes.

- Other consistency checking measures used in diagnosing dynamic systems ensure that the auxiliary assumptions are consistent, for instance, that a qualitative magnitude cannot be both positive and negative, that assumptions about

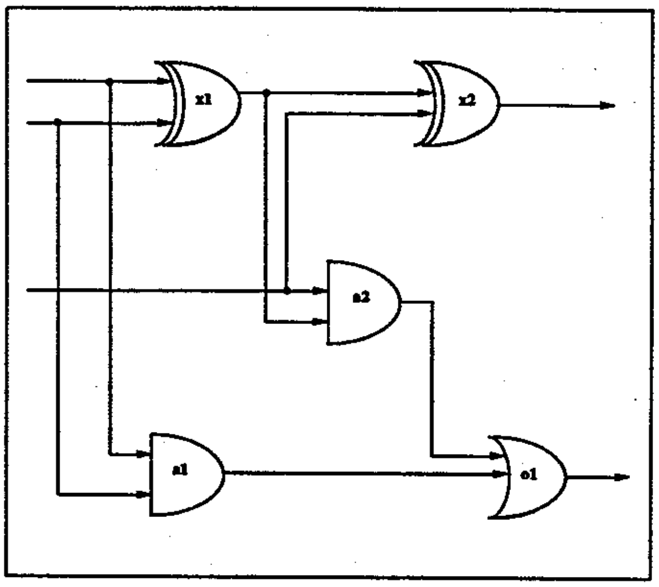

Figure 5.1: A full adder.

corresponding magnitudes are consistent. Such consistency checks are accomplished by using a pre-determined list of nogoods in the knowledge base, and by procedural code.

5.3 Diagnosing Logic Circuits

Much of model-based diagnosis research has involved diagnosing logic circuits [Davis, 1984; Genesereth, 1984; de Kleer and Williams, 1987; Reiter, 1987; de Kleer and Williams, 1989]. In this section, we describe how the abductive approach of ACCEL is used to diagnose a full adder which is representative of standard, combinational logic circuits.

Figure 5.1 shows a full adder which consists of 2 exclusive-or gates (x1, x2), two and gates (a1, a2), and one or gate (o1). We assume that each gate has 4 behavioral modes:

1. normal: the output bit reflects the correct gate behavior at all times;
2. stuck-at-0: the output bit is stuck at 0 regardless of the input bits;
3. stuck-at-1: the output bit is stuck at 1 regardless of the input bits; and
4. abnormal: the behavior of the gate is unconstrained.

The knowledge base axiom that describes the correct behavior of an exclusive-or gate is:

$$\begin{aligned} out(X,W,T) \leftarrow\ & xorg(X) \wedge in1(X,U,T) \wedge in2(X,V,T) \wedge \\ & norm(X) \wedge xor(U,V,W) \end{aligned}$$

The axiom asserts that if X is an exclusive-or gate ($xorg(X)$), the first input of X is U at time T ($in1(X,U,T)$), the second input of X is V at time T ($in2(X,V,T)$), X is normal ($norm(X)$), and the exclusive-or of U and V is W ($xor(U,V,W)$), then the output of X is W at time T ($out(X,W,T)$). In addition we have the facts $xor(0,0,0)$, $xor(0,1,1)$, $xor(1,0,1)$, and $xor(1,1,0)$. The axioms describing and gates and or gates are similar.

The following two axioms describe the two fault modes, *stuck-at-*0 and *stuck-at-*1, for all gates:

$$\begin{aligned} out(X,0,T) &\leftarrow in1(X,U,T) \wedge in2(X,V,T) \wedge stuck\text{-}at\text{-}0(X) \\ out(X,1,T) &\leftarrow in1(X,U,T) \wedge in2(X,V,T) \wedge stuck\text{-}at\text{-}1(X) \end{aligned}$$

Note that when a gate is assumed to be abnormal, no prediction can be made about its output bit. However, abduction requires that the observations be proved from the component behavioral mode assumptions (including the abnormality assumptions). To overcome this problem, we employ a technique used by David Poole [Poole, 1989b] to "parameterize" the abnormality assumption as follows:

$$out(X,W,T) \leftarrow in1(X,U,T) \wedge in2(X,V,T) \wedge ab(X,U,V,W,T)$$

The antecedent $ab(X,U,V,W,T)$ in the rule is to be interpreted as "X is abnormal in such a way that at time T, given input bits U and V, its output bit is W". Note that for any input bits U and V, and any output bit W, the above axiom always allows us to assume that the component is abnormal by making the assumption $ab(X,U,V,W,T)$. This axiom achieves our objective of being able to prove the output observations from the parameterized abnormality assumption $ab(X,U,V,W,T)$.

So far, the axioms given are not specific to the full adder; they are used to model the behavior of exclusive-or gates, and gates, and or gates. We now give the axioms that specify the connections among the gates in the given adder:

$$\begin{aligned} in1(a1,X,T) &\leftarrow in1(x1,X,T) \\ in2(a1,X,T) &\leftarrow in2(x1,X,T) \\ &\dots \end{aligned}$$

We also need facts that identify the five components: $xorg(x1), xorg(x2)$, etc.

In addition, in order to allow backward-chaining to terminate at the terminal input values of the full adder (these terminal input values cannot be further explained in terms of the other gate values), we have the following three axioms:

$$\begin{aligned} in1(x1,X,T) &\leftarrow given\text{-}in1(x1,X,T) \\ in2(x1,X,T) &\leftarrow given\text{-}in2(x1,X,T) \\ in1(a2,X,T) &\leftarrow given\text{-}in1(a2,X,T) \end{aligned}$$

and we let *given-in*1(...) and *given-in*2(...) be assumable atoms. They are auxiliary assumptions, and do not affect the simplicity metric of an explanation.

5.3.1 An Illustrative Example

Consider a scenario in which there are two faults in the full adder: *stuck-at-*0($a1$) and *stuck-at-*0($a2$). By an *I/O tuple*, we mean a particular combination of input and output values of the full adder. Suppose the first I/O tuple given to ACCEL consists of the following input atoms:

$in1(x1, 0, t1), in2(x1, 1, t1), in1(a2, 1, t1),$

$out(x2, 0, t1)$, and $out(o1, 0, t1)$.

There are two best explanations after ACCEL processes these input atoms:

$\{given\text{-}in1(x1, 0, t1),\ given\text{-}in2(x1, 1, t1),\ given\text{-}in1(a2, 1, t1),$

$norm(x1),\ norm(x2),\ stuck\text{-}at\text{-}0(a2),\ norm(a1),\ norm(o1)\}$

and

$\{given\text{-}in1(x1, 0, t1),\ given\text{-}in2(x1, 1, t1),\ given\text{-}in1(a2, 1, t1),$

$norm(x1),\ norm(x2),\ norm(a2),\ norm(a1),\ stuck\text{-}at\text{-}0(o1)\}$

The assumptions with predicates *given-in*1 and *given-in*2 are the auxiliary assumptions. The diagnoses that correspond to these best explanations are $\{stuck\text{-}at\text{-}0(a2)\}$ and $\{stuck\text{-}at\text{-}0(o1)\}$. Suppose additional measurements are taken on the intermediate output of the full adder:

$out(x1, 1, t1), out(a1, 0, t1)$, and $out(a2, 0, t1)$.

The best explanation after processing these input atoms becomes

$\{given\text{-}in1(x1, 0, t1),\ given\text{-}in2(x1, 1, t1),\ given\text{-}in1(a2, 1, t1),$

$norm(x1),\ norm(x2),\ stuck\text{-}at\text{-}0(a2),\ norm(a1),\ norm(o1)\}$

which corresponds to the diagnosis $\{stuck\text{-}at\text{-}0(a2)\}$.

A second I/O tuple and the corresponding intermediate values are given to ACCEL:

$in1(x1, 1, t2), in2(x1, 0, t2), in1(a2, 1, t2),$

$out(x2, 0, t2), out(o1, 0, t2),$

$out(x1, 1, t2), out(a1, 0, t2)$, and $out(a2, 0, t2)$.

The best explanation becomes

$\{given\text{-}in1(x1, 0, t1),\ given\text{-}in2(x1, 1, t1),\ given\text{-}in1(a2, 1, t1),$

$norm(x1),\ norm(x2),\ stuck\text{-}at\text{-}0(a2),\ norm(a1),\ norm(o1),$

$given\text{-}in1(x1, 1, t2),\ given\text{-}in2(x1, 0, t2),\ given\text{-}in1(a2, 1, t2)\}$

while the best diagnosis remains unchanged at $\{stuck\text{-}at\text{-}0(a2)\}$.

ACCEL is then given a third I/O tuple consisting of the following input atoms:

$in1(x1, 1, t3), in2(x1, 1, t3), in1(a2, 0, t3),$

$out(x2, 0, t3)$, and $out(o1, 0, t3)$.

At this point, the best diagnoses change to $\{stuck\text{-}at\text{-}0(a1), stuck\text{-}at\text{-}0(a2)\}$, and $\{stuck\text{-}at\text{-}0(o1), stuck\text{-}at\text{-}0(a2)\}$.

When ACCEL is given an additional intermediate value $out(x1, 0, t3)$, the best diagnoses remain unchanged. Finally, ACCEL is given the intermediate value *out*($a1, 0, t3$). The best diagnosis becomes the correct diagnosis {*stuck-at*-0($a1$), *stuck -at*-0($a2$)} at this point. The time taken for ACCEL to compute the correct diagnosis for this example is 30 seconds on a Sun Sparc 2 workstation.

5.3.2 Empirical Results

In order to assess the performance of ACCEL on diagnosing the full adder, we randomly generated 10 scenarios by assuming that the various behavioral modes of each gate occur with the following probabilities: *norm* 65%, *stuck-at*-0 15%, *stuck-at*-1 15%, and *ab* 5%. Each of the 10 scenarios that was actually generated had one or two gates that were faulty, and the scenarios included some where a gate was abnormal (*ab*). The example presented in the last section is one of the 10 scenarios tested.

For each scenario, we gave ACCEL I/O tuples where the input-output bits of the adder differed from those of a correctly functioning adder. For each I/O tuple, we first gave the three input bits and the two output bits of the adder, and then the output bits of the three gates x1, a1, and a2, in that order. For each scenario, we stopped as soon as the best diagnosis found by ACCEL is the correct diagnosis. We recorded the number of I/O tuples needed to converge on the correct diagnosis for each scenario.

On a Sun Sparc 2 workstation, ACCEL took an average of 17 seconds to identify the correct diagnosis for the 10 scenarios tested. The average number of I/O tuples needed before the correct diagnosis was found is 2.1.

5.4 Diagnosing Dynamic Systems

As we have discussed previously, much research in model-based diagnosis has focused on diagnosing static, discrete devices like logic circuits. However, many devices and biological systems are continuous and dynamic and require reasoning about changes in behavior over time. Although there has been a great deal of research on modeling and simulating such systems [Kuipers, 1986; Forbus, 1984], there have been few attempts to apply general, model-based diagnostic methods to them. The work of [Ng, 1991; Ng, 1990] attempts to address this deficiency by diagnosing dynamic systems using the consistency-based approach. In this section, we present an abductive approach to diagnosing continuous, dynamic systems. Specifically, ACCEL has been used to diagnose a proportional temperature controller (previously diagnosed by [Ng, 1991; Ng, 1990]) and the water balance system of the human kidney.

5.4.1 Representing Dynamic Behavior as Qualitative States

We adopt the representation of continuous dynamic systems used in the work of Kuipers' qualitative simulation (QSIM) [Kuipers, 1986; Kuipers, 1984]. The continuously changing behavior of a dynamic system over time is represented as a sequence of qualitative states, where a qualitative state consists of the qualitative values of the variables of the system. A qualitative value has two components: a qualitative magnitude (*qmag*) and a qualitative direction (*qdir*). A qualitative magnitude can either be a landmark value or an open interval between two landmark values, where a landmark value is a value of special significance that a variable takes

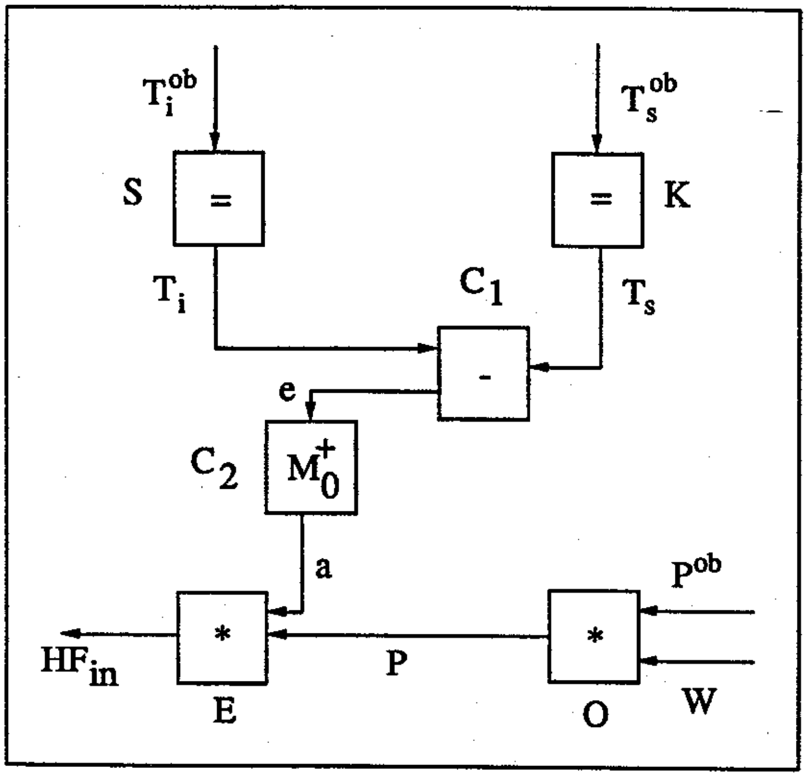

Figure 5.2: A temperature controller.

on at some point in time. A qualitative direction can be one of increasing (*inc*), decreasing (*dec*), or steady (*std*).

To illustrate the representation of dynamic behavior as qualitative states, consider a proportional temperature controller as shown in Figure 5.2. The function of this device is to control the temperature T_i^{ob} in the room, so that if the device is connected to a power source with power P^{ob}, the power switch is turned on (represented as $W = on$), and the temperature T_s^{ob} set by the temperature control knob differs from the temperature T_i^{ob} in the room, heat flow HF_{in} (in the form of hot air or cold air, depending on the direction of temperature difference) will be generated. Furthermore, the amount of heat flow generated is proportional to the temperature difference $T_s^{ob} - T_i^{ob}$.

We might observe on a cold winter day that the temperature controller is connected to the power source, the power switch is turned on, the temperature set by the control knob is at a desired room temperature, but the temperature in the room falls below the set room temperature and stabilizes at the outdoor temperature, with no heat flow generated. This observed misbehavior can be represented as three successive qualitative states s_0, s_1, and s_2, as shown in Table 5.1. In the qualitative state s_0, the qualitative value of T_i^{ob} is $\langle T_{rm}, dec \rangle$, where T_{rm} is the qualitative magnitude (the room temperature) and *dec* is the qualitative direction. In a later subsection, we will step through this faulty scenario and show how ACCEL can correctly identify the faulty components.

	s_0	s_1	s_2
T_s^{ob}	$\langle T_{rm}, std\rangle$	$\langle T_{rm}, std\rangle$	$\langle T_{rm}, std\rangle$
T_i^{ob}	$\langle T_{rm}, dec\rangle$	$\langle (Cold, T_{rm}), dec\rangle$	$\langle Cold, std\rangle$
P^{ob}	$\langle P_{sup}, std\rangle$	$\langle P_{sup}, std\rangle$	$\langle P_{sup}, std\rangle$
W	$\langle on, std\rangle$	$\langle on, std\rangle$	$\langle on, std\rangle$
HF_{in}	$\langle 0, std\rangle$	$\langle 0, std\rangle$	$\langle 0, std\rangle$

Table 5.1: Qualitative states of the faulty behavior of the temperature controller.

5.4.2 Representation of Dynamic Systems

The behavior of each dynamic system is governed by a set of qualitative constraints. Examples of qualitative constraints that we have used to model the temperature controller and the water balance system of the human kidney include:

1. $X = Y$
2. $X - Y = Z$
3. $X \cdot Y = Z$
4. $X/Y = Z$
5. $d/dt(X) = Y$
6. $m_0^+(X) = Y$

where X, Y, and Z are variables. The first four constraints represent equality, difference, multiplication, and division relations. The fifth constraint asserts that Y is the derivative of X. The last constraint asserts that there is a strictly monotonically increasing function between X and Y. However, the exact form of this monotonic function is unspecified. This accounts for the qualitative nature of the constraint.

Note that we have added some qualitative constraints $(-, /)$ in addition to those of QSIM $(+, \cdot, d/dt, m_0^+)$, although these additional constraints can be equivalently expressed using existing QSIM constraints. The reason for adding these constraints will be apparent when we discuss the logical representation of dynamic systems in ACCEL for the purpose of abductive diagnosis.

We now illustrate the use of such qualitative constraints in representing the temperature controller and the water balance system of the human kidney.

The Temperature Controller The qualitative constraints on the temperature controller are as follows (each constraint is preceded by a name identifying that constraint):

1. $S : T_i^{ob} = T_i$
2. $K : T_s^{ob} = T_s$
3. $C_1 : T_s - T_i = e$

4. $C_2 : m_0^+(e) = a$
5. $O : P^{ob} \cdot W = P$
6. $E : a \cdot P = HF_{in}$

The first constraint reflects the fact that if the temperature sensor is working normally, then the observed temperature in the room (T_i^{ob}) is the same as the temperature actually recorded by the sensor (T_i). Similarly, the second constraint asserts that if the temperature control knob is working normally, then the observed temperature set by the knob (T_s^{ob}) is the same as the temperature actually recorded by the knob (T_s). The third and fourth constraints require a normal working proportional controller to determine the adjustment amount (a) that regulates heat flow to be proportional to the difference (e) between the temperature set (T_s) and the temperature in the room (T_i). The fifth constraint implies that if the on/off power switch is functioning properly, then the power input (P) to the electric heating/cooling component is the same as the product of the observed power input (P^{ob}) and the on/off control W, where $W = on(1)$ if the switch is on and $W = off(0)$ otherwise. The last constraint requires the heating/cooling component to output the correct amount of heat flow (HF_{in}) according to the product of the adjustment amount (a) and the power supply (P).

The Water Balance System of the Human Kidney The water balance system of the human kidney is based around anti-diuretic hormone (ADH) [Kuipers, 1985; Kuipers, 1991]. The qualitative constraints of the system are (each constraint is preceded by a name identifying that constraint):

1. $C_1 : amt(Na, P)/amt(water, P) = c(Na, P)$
2. $C_2 : m_0^+(amt(water, P)) = c(NH, P)$
3. $C_3 : m_0^+(c(NH, P)) = flow(P\text{-}U)$
4. $C_4 : m_0^+(c(Na, P)) = c(ADH, P)$
5. $C_5 : m_0^+(c(ADH, P)) = flow(U\text{-}P)$
6. $C_6 : flow(P\text{-}U) - flow(U\text{-}P) = netflow(P\text{-}U)$
7. $C_7 : netflow(in\text{-}P) - netflow(P\text{-}U) = netflow(out\text{-}P)$
8. $C_8 : d/dt(amt(water, P)) = netflow(out\text{-}P)$

In this system, two variables are essentially constant (when considered on the time-scale that the system takes to reach equilibrium):

1. the amount of sodium in the plasma, $amt(Na, P)$, and
2. the rate of water intake, $netflow(in\text{-}P)$.

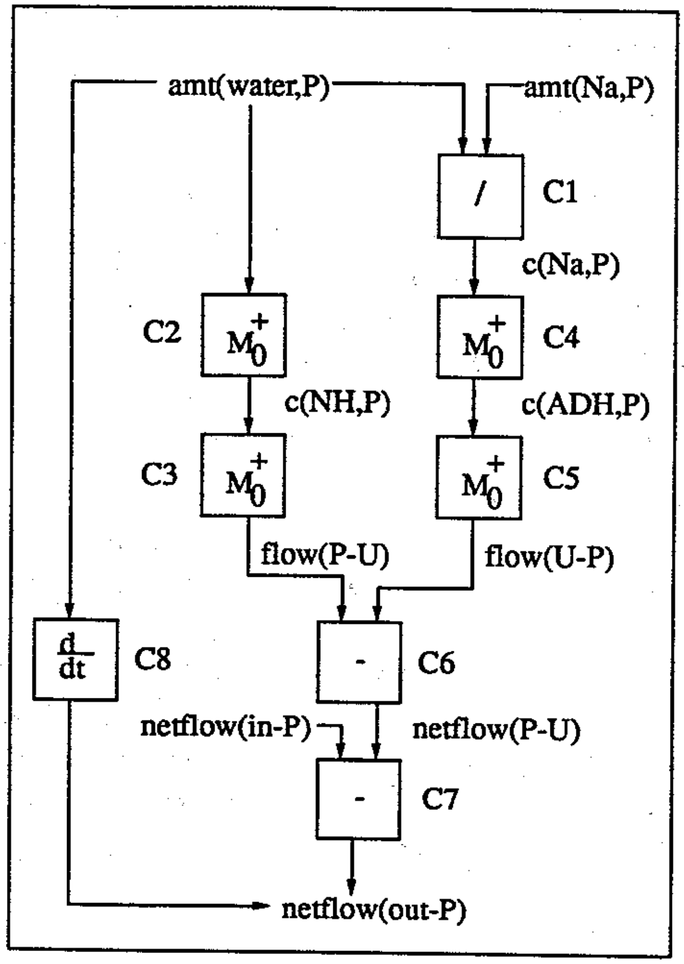

Figure 5.3: The water balance system of the human kidney.

The first constraint asserts that the concentration of sodium in the plasma ($c(Na, P)$) is the amount of sodium in the plasma ($amt(Na, P)$) divided by the amount of water in the plasma ($amt(water, P)$). The second constraint asserts that the amount of water in the plasma affects monotonically the concentration of natriuretic hormone in the plasma ($c(NH, P)$), which in turn affects monotonically the flow of water from the plasma into the nephron ($flow(P\text{-}U)$) via the third constraint. Similarly, the fourth constraint asserts that the concentration of sodium in the plasma affects monotonically the concentration of ADH in the plasma ($c(ADH, P)$), which in turn affects the flow of water reabsorbed from the nephron back into the plasma ($flow(U\text{-}P)$) via the fifth constraint. The sixth constraint asserts that the netflow of water from the plasma into the nephron ($netflow(P\text{-}U)$) is the difference between the flow of water from the plasma into the nephron and the flow of water reabsorbed in the other direction. The seventh constraint asserts that the rate of water intake ($netflow(in\text{-}P)$) minus the netflow rate of water from the plasma into the nephron is the overall netflow rate of water into the plasma ($netflow(out\text{-}P)$). The last constraint asserts that the rate of change of the amount of water in the plasma is the overall netflow rate of water into the plasma.

Given knowledge about the structure of a dynamic system, the constraints governing its correct behavior, and its known fault modes, the diagnostic task is to identify the failing components from the qualitative states that represent the device misbehavior over time.

5.4.3 Horn-Clause Representation of Qualitative Constraints

We have successfully represented QSIM's knowledge about the various qualitative constraints $(=, -, \cdot, /, d/dt, m_0^+)$ in Horn-clause axioms in a way suitable for logic-based abductive diagnosis. Since these Horn-clause axioms encode general knowledge about QSIM constraints, they are needed in the diagnosis of every dynamic system. These axioms encode the various qualitative constraints by defining a "holds.constraint-type" predicate for each type of qualitative constraint.

For example, the following 9 axioms encode the m_0^+ constraint:

$$\begin{array}{rcl}
holds.m_0^+(F, G, M1, inc, M2, inc) & \leftarrow & pos(M1) \wedge pos(M2) \wedge \\
 & & corr\text{-}mag.m_0^+(F, G, M1, M2) \\
holds.m_0^+(F, G, M1, std, M2, std) & \leftarrow & pos(M1) \wedge pos(M2) \wedge \\
 & & corr\text{-}mag.m_0^+(F, G, M1, M2) \\
holds.m_0^+(F, G, M1, dec, M2, dec) & \leftarrow & pos(M1) \wedge pos(M2) \wedge \\
 & & corr\text{-}mag.m_0^+(F, G, M1, M2) \\
holds.m_0^+(F, G, M1, inc, M2, inc) & \leftarrow & neg(M1) \wedge neg(M2) \wedge \\
 & & corr\text{-}mag.m_0^+(F, G, M1, M2) \\
holds.m_0^+(F, G, M1, std, M2, std) & \leftarrow & neg(M1) \wedge neg(M2) \wedge \\
 & & corr\text{-}mag.m_0^+(F, G, M1, M2) \\
holds.m_0^+(F, G, M1, dec, M2, dec) & \leftarrow & neg(M1) \wedge neg(M2) \wedge \\
 & & corr\text{-}mag.m_0^+(F, G, M1, M2) \\
holds.m_0^+(F, G, 0, inc, 0, inc) & & \\
holds.m_0^+(F, G, 0, std, 0, std) & & \\
holds.m_0^+(F, G, 0, dec, 0, dec) & &
\end{array}$$

$holds.m_0^+(F, G, M1, D1, M2, D2)$ asserts that $m_0^+(F) = G$ holds with the qualitative value of the variable $F = \langle M1, D1 \rangle$ and the qualitative value of the variable $G = \langle M2, D2 \rangle$. $pos(M1)$ ($neg(M1)$) asserts that the qualitative magnitude $M1$ is positive (negative). $corr\text{-}mag.m_0^+(F, G, M1, M2)$ asserts that $m_0^+(F) = G$ holds with the qualitative magnitude of $F = M1$ and the qualitative magnitude of $G = M2$. In QSIM, $(M1, M2)$ are referred to as corresponding values. These 9 axioms cover all the distinct possibilities in which $m_0^+(F) = G$ holds since the qualitative magnitude of F can be positive, negative, or zero, and its qualitative direction can be *inc*, *std*, or *dec*.

Other "holds.constraint-type" predicates include *holds.–*, *holds.∗*, *holds./*, and *holds.d/dt*. Atoms with one of these "holds.constraint-type" predicates are not assumable. Hence, when ACCEL encounters a subgoal with a $holds.m_0^+$ predicate, it will be forced to pursue each of the 9 alternatives (backward-chain on the first six

F	G	$H = F - G$
inc	*inc*	*inc*
inc	*inc*	*std*
inc	*inc*	*dec*
inc	*std*	*inc*
inc	*dec*	*inc*
std	*inc*	*dec*
std	*std*	*std*
std	*dec*	*inc*
dec	*inc*	*dec*
dec	*std*	*dec*
dec	*dec*	*inc*
dec	*dec*	*std*
dec	*dec*	*dec*

Table 5.2: Valid combinations of qualitative directions.

rules or unify with the last three facts). Atoms with the predicates *pos*, *neg*, and *corr-mag.*m_0^+ are assumable, and they form the "auxiliary" assumptions in an abductive explanation. The auxiliary assumptions *pos*(M) and *neg*(M) make explicit the sign of the qualitative magnitude M, while the assumption *corr-mag.*$m_0^+(F, G, M1, M2)$ enforces the corresponding values constraint in QSIM which asserts that the two qualitative magnitudes $M1$ and $M2$ are valid corresponding values for the constraint $m_0^+(F) = G$ (i.e., when *qmag*$(F) = M1$, *qmag*$(G) = M2$). In addition, consistency checking in ACCEL uses a list of known corresponding values for m_0^+ constraints to ensure that if $(M1, M2)$ are given as corresponding values for the constraint $m_0^+(F) = G$, then the assumption *corr-mag.*$m_0^+(F, G, M1, M3)$ (or *corr-mag.*$m_0^+(F, G, M3, M2)$) is inconsistent for $M2 \neq M3$ (or $M1 \neq M3$).

Similarly, *holds.*$-(F, G, H, M1, D1, M2, D2, M3, D3)$ asserts that $F - G = H$ holds with $F = \langle M1, D1\rangle$, $G = \langle M2, D2\rangle$, and $H = \langle M3, D3\rangle$. There are 39 axioms defining *holds.* $-(F, G, H, M1, D1, M2, D2, M3, D3)$, and they can be divided into three equal groups, corresponding to the cases: (1) *qmag*$(F) =$ *qmag*(H), *qmag*$(G) = 0$; (2) *qmag*$(F) =$ *qmag*(G), *qmag*$(H) = 0$; and (3) *qmag*$(F) \neq$ *qmag*(H), *qmag*$(F) \neq$ *qmag*(G). The 13 axioms in each group cover all the different possible combinations of qualitative directions of F, G, and H that obey the first-order derivative constraint $F' - G' = H'$. These valid combinations of qualitative directions are shown in Table 5.2.

The remaining "holds.constraint-type" predicates, *holds.*$*$, *holds./*, and *holds.d/dt*, are defined by 97, 70, and 9 axioms, respectively. The axioms for *holds.*$*$ $(F, G, H, M1, D1, M2, D2, M3, D3)$ ensure that, among other things, the first-order derivative constraint $F \cdot G' + F' \cdot G = H'$ is obeyed. The exact axioms for all the qualitative constraints are listed in Appendix C.

5.4.4 Horn-Clause Representation of Dynamic Systems

Besides the above axioms that encode general QSIM qualitative constraints, there are also Horn-clause axioms that encode knowledge about a specific dynamic system. Our Horn-clause representation and diagnosis of a dynamic system is done at the constraint level, and we assume that each constraint governs the behavior of one component of a system. Also, we assume that when a dynamic system malfunctions, it is due to one or more components whose behavior violate their respective constraints. Diagnosis at the constraint level (as opposed to the component level) is at a more refined level of granularity. This is because if the behavior of some component C is governed by more than one qualitative constraints, we can always assume that each constraint corresponds to a mini-component c_i, so that if any c_i is found to be faulty, C is also faulty.

The Temperature Controller The following axioms describe the normal behavior of each constraint/component of the temperature controller:

$$
\begin{aligned}
qval(ti, M1, D1, T) &\leftarrow norm(s) \wedge qval(ti\text{-}ob, M1, D1, T) \\
qval(ts, M1, D1, T) &\leftarrow norm(k) \wedge qval(ts\text{-}ob, M1, D1, T) \\
qval(e, M3, D3, T) &\leftarrow norm(c1) \wedge \\
&\quad\; qval(ts, M1, D1, T) \wedge qval(ti, M2, D2, T) \wedge \\
&\quad\; holds.-(ts, ti, e, M1, D1, M2, D2, M3, D3) \\
qval(a, M2, D1, T) &\leftarrow norm(c2) \wedge qval(e, M1, D1, T) \wedge \\
&\quad\; holds.m_0^+(e, a, M1, D1, M2, D1) \\
qval(p, M3, D3, T) &\leftarrow norm(o) \wedge \\
&\quad\; qval(p\text{-}ob, M1, D1, T) \wedge qval(w, M2, D2, T) \wedge \\
&\quad\; holds.*(p\text{-}ob, w, p, M1, D1, M2, D2, M3, D3) \\
qval(hfin, M3, D3, T) &\leftarrow norm(e) \wedge \\
&\quad\; qval(a, M1, D1, T) \wedge qval(p, M2, D2, T) \wedge \\
&\quad\; holds.*(a, p, hfin, M1, D1, M2, D2, M3, D3)
\end{aligned}
$$

$qval(ti, M1, D1, T)$ asserts that the qualitative value of the variable ti is $\langle M1, D1\rangle$ at time (qualitative state) T. The first axiom asserts that if component s is normal, and the qualitative value of *ti-ob* is $\langle M1, D1\rangle$ at time T, then the qualitative value of ti is also $\langle M1, D1\rangle$ at time T. This encodes the equality constraint between the variables *ti-ob* and ti. The second axiom similarly encodes the equality constraint between *ts-ob* and ts. The third axiom asserts that if component c1 is normal, the qualitative value of ts is $\langle M1, D1\rangle$ at time T, the qualitative value of ti is $\langle M2, D2\rangle$ at time T, and $ts - ti = e$ holds with $ts = \langle M1, D1\rangle, ti = \langle M2, D2\rangle, e = \langle M3, D3\rangle$, then the qualitative value of e is $\langle M3, D3\rangle$ at time T. The remaining three axioms are interpreted in a similar way.

Note that atoms with the predicate *qval* are not assumable. As such, in order to allow backward-chaining to terminate at the terminal input values of a dynamic device (these terminal input values cannot be further explained), we also need the following axioms:

$$
\begin{aligned}
qval(ti\text{-}ob, M1, D1, T) &\leftarrow given\text{-}qval(ti\text{-}ob, M1, D1, T) \\
qval(ts\text{-}ob, M1, D1, T) &\leftarrow given\text{-}qval(ts\text{-}ob, M1, D1, T) \\
qval(p\text{-}ob, M1, D1, T) &\leftarrow given\text{-}qval(p\text{-}ob, M1, D1, T) \\
qval(w, M1, D1, T) &\leftarrow given\text{-}qval(w, M1, D1, T)
\end{aligned}
$$

and we let *given-qval*(...) be assumable. They are part of the "auxiliary" assumptions in an abductive explanation.

Note the directionality in which one qualitative value is explained in terms of the other qualitative values. Since abductive diagnosis requires that the input observations (which consists of the qualitative values of the variables of a dynamic system) be *proved*, the axioms are formulated in such a way that the output values (e.g., $qval(hfin, \ldots)$) of a dynamic system can be proved from normality assumptions (e.g., $norm(s)$), fault mode assumptions, and auxiliary assumptions about the input values (e.g., $given\text{-}qval(ti\text{-}ob, \ldots)$) and the qualitative magnitudes and corresponding values of the variables (these are introduced when ACCEL attempts to prove the holds.constraint-type atoms). The need to express the axioms in the proper causal direction is the reason we have added the constraint types $-$ and $/$ in addition to those qualitative constraints already present in QSIM. In addition, as long as the qualitative constraints are formulated in such a way that output variables are expressed in terms of input variables, the Horn-clause axioms in ACCEL that represent the qualitative constraints of a particular dynamic system (i.e., those axioms listed in this subsection) can be automatically generated from their description in equational form (i.e., those presented in subsection 5.4.2).

We also assume that the various components of the temperature controller have the following fault modes:

- S: stuck-at-0-std, stuck-at-roomtemp-std;
- K: stuck-at-0-std;
- C_1: stuck-at-0-std, stuck-at-1st-in, stuck-at-2nd-in;
- C_2: stuck-at-0-std;
- O: stuck-at-0-std, stuck-at-1st-in; and
- E: stuck-at-0-std.

Under the fault mode stuck-at-0-std (stuck-at-roomtemp-std), the output of a component is $\langle 0, std\rangle$ ($\langle room\text{-}temp, std\rangle$) regardless of the input values. Under the fault mode stuck-at-1st-in (stuck-at-2nd-in), the output of a component is stuck at its first (second) input. One Horn-clause axiom is used to encode one fault mode of a component, as follows:

$$\begin{aligned}
qval(ti, 0, std, T) &\leftarrow stuck\text{-}at\text{-}0\text{-}std(s) \wedge \\
&\quad qval(ti\text{-}ob, M1, D1, T) \\
&\ldots \\
qval(ti, roomtemp, std, T) &\leftarrow stuck\text{-}at\text{-}roomtemp\text{-}std(s) \wedge \\
&\quad qval(ti\text{-}ob, M1, D1, T) \wedge pos(roomtemp) \\
qval(e, M1, D1, T) &\leftarrow stuck\text{-}at\text{-}1st\text{-}in(c1) \wedge \\
&\quad qval(ts, M1, D1, T) \wedge qval(ti, M2, D2, T) \\
qval(e, M2, D2, T) &\leftarrow stuck\text{-}at\text{-}2nd\text{-}in(c1) \wedge \\
&\quad qval(ts, M1, D1, T) \wedge qval(ti, M2, D2, T) \\
qval(p, M1, D1, T) &\leftarrow stuck\text{-}at\text{-}1st\text{-}in(o) \wedge \\
&\quad qval(p\text{-}ob, M1, D1, T) \wedge qval(w, M2, D2, T)
\end{aligned}$$

The Water Balance System of the Human Kidney Similar axioms are used to represent the kidney water balance system. Note that our Horn-clause axioms only model the seven constraints $C_1, \ldots, C_7$. We assume that the constraint $C_8 : d/dt(amt(water, P)) = netflow(out\text{-}P)$ is not violated and do not represent it in the axioms. This is to avoid causal loops since otherwise $netflow(out\text{-}P)$ can be explained in terms of $amt(water, P)$ via the constraints C_7, C_6, C_3, and C_2 but $amt(water, P)$ can be explained in terms of $netflow(out\text{-}P)$ via the constraint C_8. The fault modes of the system include:

- C_2: stuck-at-low-std, stuck-at-high-std;
- C_3: stuck-at-low-std, stuck-at-high-std;
- C_4: stuck-at-low-std, stuck-at-high-std; and
- C_5: stuck-at-low-std, stuck-at-high-std.

The axioms for the fault modes are:

$$\begin{aligned} qval(cnh, cnh-, std, T) &\leftarrow stuck\text{-}at\text{-}low\text{-}std(c2) \wedge \\ &\qquad qval(awp, M1, D1, T) \wedge pos(cnh-) \\ qval(cnh, cnh+, std, T) &\leftarrow stuck\text{-}at\text{-}high\text{-}std(c2) \wedge \\ &\qquad qval(awp, M1, D1, T) \wedge pos(cnh+) \\ &\ldots \end{aligned}$$

awp and *cnh* are abbreviations for $amount(water, P)$ and $c(NH, P)$, respectively. *stuck-at-high-std*(*c*4) or *stuck-at-high-std*(*c*5) corresponds to the Syndrome of Inappropriate Secretion of Anti-Diuretic hormone (SIADH), while *stuck-at-low-std*(*c*4) or *stuck-at-low-std*(*c*5) corresponds to Diabetes Insipidus [Kuipers, 1985; Kuipers, 1991].

The Horn-clause axioms in ACCEL that represent the qualitative constraints capture the knowledge that QSIM uses to propagate qualitative values across constraints in order to complete the qualitative values of variables in a qualitative state. In ACCEL, such knowledge is used for the purpose of diagnosis. However, since the knowledge is now encoded declaratively, it can also be used for simulation purpose by a forward-chaining inference procedure. In fact, QSIM can be viewed as a special-purpose theorem prover for predicting the behavior of dynamic systems described by qualitative constraints. However, not all of QSIM's knowledge in simulation has been captured in ACCEL. Specifically, knowledge of continuity constraints that QSIM uses to generate the next qualitative state(s) from an initial qualitative state is not encoded in ACCEL, since such knowledge is not needed in diagnosis.

5.4.5 An Illustrative Example

Consider the faulty scenario described in subsection 5.4.1, in which the temperature controller is connected to the power source, the power switch is turned on, the temperature set by the control knob is at a desired room temperature, but the temperature in the room falls below the set room temperature and stabilizes at the outdoor temperature, with no heat flow generated into the room. Suppose that in this case, there are two faults present in the device, *stuck-at-*0*-std*(*s*) and *stuck-at-*0*-std*(*c*2).

Suppose the following input atoms representing the initial qualitative state are given to ACCEL: *pos(roomtemp)*, *qval(ts-ob, roomtemp, std, t1)*, *qval(ti-ob, roomtemp, dec, t1)*, *pos(psup)*, *qval(p-ob, psup, std, t1)*, *pos(on)*, *qval(w, on, std, t1)*, and *qval(hfin, 0, std, t1)*. After incrementally processing each of these input atoms, the best diagnoses (those with the least number of components that are not normal) computed by ACCEL are:

1. {*stuck-at-roomtemp-std(s)*},
2. {*stuck-at-0-std(c1)*},
3. {*stuck-at-0-std(c2)*},
4. {*stuck-at-0-std(o)*}, and
5. {*stuck-at-0-std(e)*}.

Note that the abductive explanation corresponding to each of the diagnoses includes the behavioral mode assumptions of each component of the device, as well as the auxiliary assumptions. For instance, the abductive explanation corresponding to the third diagnosis has the following assumptions:

$$\{norm(s), norm(k), norm(c1), stuck\text{-}at\text{-}0\text{-}std(c2), norm(o), norm(e),$$
$$pos(roomtemp), given\text{-}qval(ts\text{-}ob, roomtemp, std, t1),$$
$$given\text{-}qval(ti\text{-}ob, roomtemp, dec, t1), pos(psup),$$
$$given\text{-}qval(p\text{-}ob, psup, std, t1), pos(on), given\text{-}qval(w, on, std, t1),$$
$$pos(P16), corr\text{-}mag.{*}(p\text{-}ob, w, p, psup, on, P16)\}.$$

The auxiliary assumptions include postulating a positive qualitative magnitude *P*16 that is the product of *psup* and *on*.

Next, we consider the qualitative state corresponding to the immediately following distinguished time point where the temperature in the room stabilizes at the outdoor temperature. Assume that the next input atoms consist of the qualitative values of the input and output variables at this qualitative state: *qval(ts-ob, roomtemp, std, t2)*, *pos(cold)*, *>(roomtemp, cold)*, *qval(ti-ob, cold, std, t2)*, *qval(p-ob, psup, std, t2)*, *qval(w, on, std, t2)*, and *qval(hfin, 0, std, t2)*. The best diagnoses remain unchanged after processing these input atoms.

Now, suppose we start measuring the qualitative values of the intermediate variables of the device at this quiescent state. (For simplicity, we assume that all intermediate variables are measurable.) Suppose the next input atom given to ACCEL is *qval(ts, roomtemp, std, t2)*, which is the temperature actually recorded by the control knob. Again, the best diagnoses remain unchanged.

Next, ACCEL processes the input atom *qval(ti, 0, std, t2)*, which gives the temperature actually recorded by the temperature sensor. The best diagnoses now change to:

1. {*stuck-at-0-std(s)*, *stuck-at-0-std(c1)*},
2. {*stuck-at-0-std(s)*, *stuck-at-2nd-in(c1)*},
3. {*stuck-at-0-std(s)*, *stuck-at-0-std(c2)*}, and

4. $\{stuck\text{-}at\text{-}0\text{-}std(s), stuck\text{-}at\text{-}0\text{-}std(e)\}$.

ACCEL continues to process the next input atom $qval(e, roomtemp, std, t2)$ which gives the temperature difference recorded by the controller. The best diagnoses now become:

1. $\{stuck\text{-}at\text{-}0\text{-}std(s), stuck\text{-}at\text{-}0\text{-}std(c2)\}$, and
2. $\{stuck\text{-}at\text{-}0\text{-}std(s), stuck\text{-}at\text{-}0\text{-}std(e)\}$.

Finally, ACCEL processes the input atom $qval(a, 0, std, t2)$ (the adjustment amount) and the best diagnosis returned is the correct diagnosis $\{stuck\text{-}at\text{-}0\text{-}std(s), stuck\text{-}at\text{-}0\text{-}std(c2)\}$. The time taken for ACCEL to arrive at the correct diagnosis for this scenario is 3.99 minutes.

5.4.6 Empirical Results

We tested ACCEL on 10 scenarios each on the temperature controller and the water balance system of the human kidney. Measurements of intermediate variables are made in a fixed order so as to further differentiate and narrow the diagnostic candidates. Although our work does not focus on experimentation, the ability to intelligently select which component output and which additional sensor values to measure is important in achieving efficient diagnosis.

The Temperature Controller In order to generate the faulty scenarios for the temperature controller, we used a QSIM model (Figure 5.4) of the temperature controller embedded within its surrounding environment [Kuipers, 1991]. This model contains qualitative constraints for the heat flow both between the heating/cooling element of the temperature controller and the room, and between the room and the outdoor environment.

We randomly generated 10 scenarios where each scenario contains one to two faults and in which no heat flow was generated into the room. For each scenario, we gave the input atoms representing the qualitative values of the following variables in the order listed:

1. $T_s^{ob}, T_i^{ob}, P^{ob}, W, HF_{in}$ at the initial qualitative state (t_1);
2. $T_s^{ob}, T_i^{ob}, P^{ob}, W, HF_{in}$ at the next distinguished time-point qualitative state (t_2); and
3. the intermediate variables T_s, T_i, e, a, P at state t_2.

In 9 out of the 10 scenarios, ACCEL found the correct diagnosis as its best diagnosis. The scenario that ACCEL failed to find the best diagnosis has two faults $\{stuck\text{-}at\text{-}0\text{-}std(c1), stuck\text{-}at\text{-}0\text{-}std(c2)\}$. In this case, the best diagnosis that ACCEL found after processing all the intermediate variables is $\{stuck\text{-}at\text{-}0\text{-}std(c1)\}$. This is as it should be, since when $c1$ is stuck at $\langle 0, std\rangle$, the correct behavior of $c2$ if it is normal is to output $a = \langle 0, std\rangle$ at all times, which is indistinguishable from the behavior of $c2$ if it is in the fault mode *stuck-at-0-std*. That $c2$ is in fact faulty would be detected when $c1$ is replaced by a normal, working component and the controller is still found to be malfunctioning.

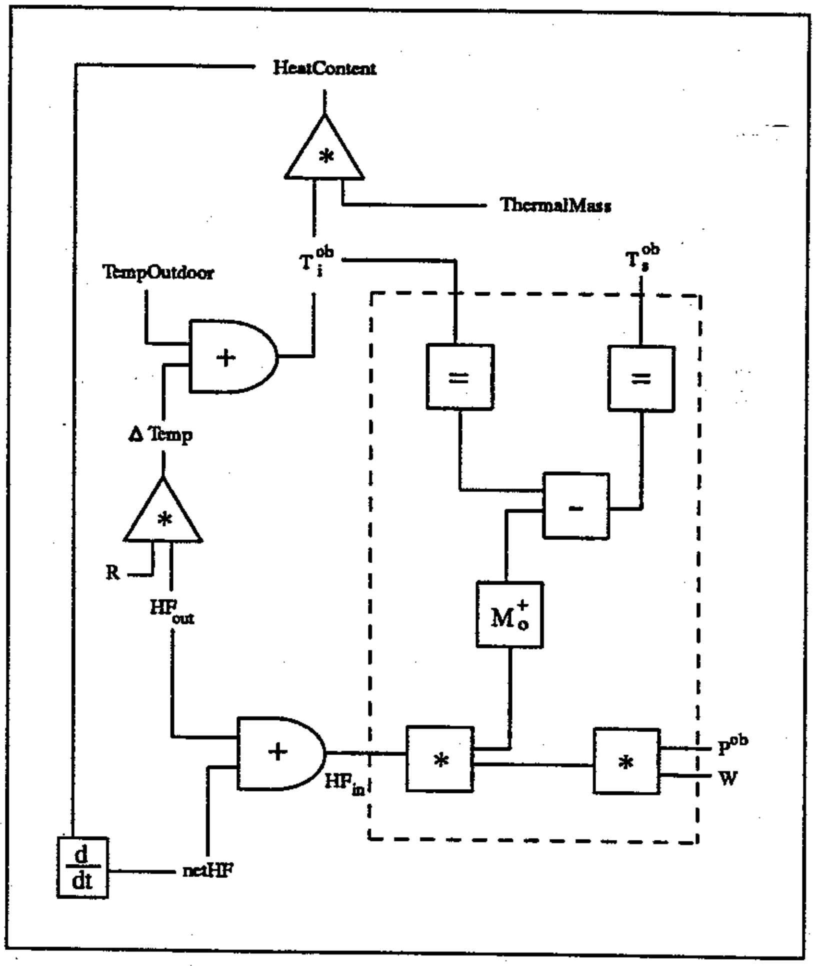

Figure 5.4: The temperature controller and its environment.

Overall, the average run time per scenario is 4.24 minutes, and the average number of measurements of intermediate variables needed to arrive at the correct diagnosis is 4.4.

The Water Balance System of the Human Kidney We also tested ACCEL on 10 faulty scenarios of the kidney water balance system. The 10 scenarios consist of all the 8 possibilities that one of the constraints $C_2, \ldots, C_5$ is stuck at low/high, and two scenarios that have 2 faults. Each of the 10 scenarios starts with an initial qualitative state with high water intake into the body, and then reaches equilibrium over time. For each of the 1-fault scenarios, we gave the input atoms representing the qualitative values of the following variables in the order listed:

1. $amt(Na, P), amt(water, P), netflow(in\text{-}P), netflow(out\text{-}P)$ at the initial qualitative state (t_1);

Problem	Time (min)			Inference count		
	No-cache	Cache	speedup	No-cache	Cache	ratio
train1	22.19	3.10	7.16	1418	138	10.27
train8	0.43	0.24	1.79	118	45	2.62
train13	11.42	2.02	5.65	919	120	7.66
test9	>25.04	2.96	>8.46	>1401	127	>11.03
test19	>26.78	4.17	>6.42	>1561	209	>7.47
brain5	0.16	0.07	2.29	2198	142	15.48
brain10	0.07	0.03	2.33	1004	63	15.94
brain42	0.05	0.02	2.50	783	100	7.83
adder1	43.77	0.51	85.82	157210	749	209.89
adder4	3.81	0.08	47.63	30225	237	127.53
adder10	4.79	0.12	39.92	38110	255	149.45
tc1	>72.53	4.70	>15.43	>17766	442	>40.19
tc4	>68.08	4.35	>15.65	>17766	415	>42.81
tc8	>68.22	4.26	>16.01	>17766	385	>46.15
kidney1	>61.24	7.86	>7.79	>8661	470	>18.42
kidney5	>65.26	7.89	>8.27	>8577	433	>19.81
Average			17.07			45.78

Table 6.1: Empirical results comparing caching and non-caching performance

architecture, creates rules to summarize the processing of a subgoal so that future problem solving in the subgoal can be avoided [Laird *et al.*, 1984].

Previous research on caching has sometimes produced conflicting evidence as to the utility of caching. Although Elkan achieved good results with the use of caching in [Elkan, 1989], he also reported that Stickel had independently discovered and implemented a caching scheme similar to his, but that the results Stickel obtained were unfavorable to caching on the class of theorems Stickel investigated at the time. This is analogous to the *utility problem* in explanation-based learning [Minton, 1988], where learning more search control knowledge may or may not improve the overall performance of a system.

As mentioned in Chapter 2, we believe that duplicating inference poses a more serious problem in abduction because multiple abductive proofs must usually be pursued in the search for a best explanation, whereas in deduction, we are usually interested in a single deductive proof. The need for multiple abductive proofs tends to result in more duplicate inferences being made, since the multiple abductive proofs maintained tend to share many identical subgoals. In [Stickel, 1991], Stickel has also expressed similar opinions of the "strong motivation" to "eliminate search-space redundancy" for abduction since "the presence of an additional inference rule that allows literals to be either assumed or proved makes the search space for abduction even larger than that for deduction". Our empirical results confirm that caching is indeed very effective in improving the efficiency of abduction.

Chapter 7

Related Work

There is a great deal of research related to abduction, plan recognition, and diagnosis. Previous general abduction algorithms and systems are too restrictive, too inefficient, and not well tested on real problems. On the other hand, related research in the areas of plan recognition and diagnosis are too domain specific and not based on a general, unifying formalism. For example, the generalized set covering model is propositional and is not applicable to modeling plan recognition in narrative texts which require the expressive power of first-order predicate logic. Another example is the ATMS algorithm which only deals with propositional abduction. Our discussions of related work are divided into the following subsections: general theory and algorithms, plan recognition and natural language understanding, diagnosis, and abduction in other domains.

7.1 General Theory and Algorithms

Pople was the first researcher to explore abductive reasoning in AI [Pople, 1973], although he was mainly concerned with using abduction to perform disease diagnosis. Charniak and McDermott proposed abduction as a general model for explanation, and recognized that many diverse AI tasks, including natural language understanding, diagnosis, and image interpretation, can be elegantly modeled as abduction [Charniak and McDermott, 1985]. Our work takes this hypothesis one step further and demonstrates via an implemented system that general and efficient abduction for the tasks of plan recognition and diagnosis is indeed possible.

The SAA algorithm given in Chapter 2 is a general abduction algorithm for first-order Horn-clause domain theories [Stickel, 1988a]. However, it does not perform caching of partial explanations and therefore duplicates inferences. To address this problem, Stickel has proposed a method to formulate a goal-directed, backward-chaining algorithm "metatheoretically" for execution by a forward-chaining reasoning system such as hyperresolution [Stickel, 1991]. That is, metapredicates such as *fact* and *goal* are used to encode a backward-chaining algorithm using axioms to be executed by a forward-chaining inference system. In this way, subsumption checks in the forward-chaining system will ensure that duplicate inferences are not made, and the goal-directedness of the backward-chaining algorithm can also be preserved. Such an approach has the advantage that it can be easily implemented on an existing high-performance, forward-chaining system (such as OTTER [McCune, 1990]). Our AAA algorithm achieves analogous effects of goal-directedness and non-duplicating inference in a direct way, via backward-chaining and caching.

Levesque has characterized abduction in terms of a formal model of belief [Levesque, 1989]. He proved that for propositional Horn-clause theories, the ATMS [de Kleer, 1986] is an abductive procedure that computes all minimal explanations. Our AAA algorithm is more general and computes first-order Horn-clause abductive proofs.

Ginsberg has implemented a first-order ATMS using a multi-valued-logic theorem prover, MVL [Ginsberg, 1989]. Compared to Accel, MVL is a more general theorem prover for full first-order predicate logic and it is capable of many kinds of reasoning including default reasoning, circumscription, temporal reasoning, and probabilistic reasoning. However, his implementation of the first-order ATMS does not cache previously computed partial explanations. This is in marked contrast to the (propositional) ATMS of de Kleer, in which caching and sharing of explanations are the distinguishing features. Hence, Ginsberg's system is an "ATMS" only in the sense that it is an algorithm that computes all possible proofs (explanations). In addition, his system has not been tested on large problems.

Kautz has developed a formal theory of plan recognition based on first-order predicate logic [Kautz, 1987; Kautz and Allen, 1986]. In his theory, an event hierarchy captures isa relationships between events (abstraction hierarchy) as well as part-of relationships of events and their components (decomposition hierarchy). He uses the notion of a minimum covering model to define mc-entailment, where an observation mc-entails a conclusion if and only if the conclusion deductively follows from the observation, the event hierarchy, completeness assumptions about the event hierarchy, and the assumption that as few top-level events occur as possible. It is shown that mc-entailment is related to circumscription [McCarthy, 1980]. The assumption that as few top-level events occur as possible is a form of simplicity measure. This is in contrast to our use of coherence metric, which we have shown to be a better metric than simplicity in the plan recognition domain.

One major difference between Kautz's theory and our work is that his theory models plan recognition as non-monotonic deduction rather than abduction. The difference is similar to that between consistency-based diagnosis and abductive diagnosis. Kautz's theory of plan recognition is in the style of consistency-based theories and relies on making completeness assumptions about the event hierarchy as well as the the assumption that as few top-level events occur as possible. These additional assumptions enable the deductive inference of plans from observed actions. On the contrary, our abductive model of plan recognition assumes the existence of top-level events and other (e.g., role-filler) assumptions in order to prove the observed actions. In addition, due to considerations of computational efficiency, we restrict our knowledge base to Horn clauses, whereas Kautz allowed arbitrary first-order formulas.

The computational complexity of several abductive problems has been formally analyzed. It has been shown that, even in the propositional case, computing all minimal explanations is provably exponential [McAllester, 1985; Selman and Levesque, 1990], since in the worst case, the number of minimal explanations is exponentially large. Reggia *et. al.* have shown that finding parsimonious (i.e., minimum) explanations in the GSC model is NP-hard [Reggia *et al.*, 1985]. Bylander *et. al.* have investigated the complexity of various classes of abduction in which abduction is characterized as a problem of finding the most plausible composite hypothesis (a conjunction of individual hypotheses) that explains all of the data [Bylander *et al.*, 1989]. The classes of abduction are differentiated by the way hypotheses interact. They show that unless some very restrictive conditions are satisfied, abduction is computationally intractable.

Note that the GSC model and the various classes of abduction studied by Reggia *et. al.* and Bylander *et. al.* only concern propositional abduction in which abductive proofs are restricted to be of depth one. Similarly, Levesque's characterization of abduction is in terms of propositional beliefs. However, our abduction model is more general in that it allows first-order Horn clause axioms with variables.

To limit the computational efforts expended in the ATMS, Forbus and de Kleer introduced a "focusing" technique in which only relevant environments in an ATMS are maintained and propagated [Forbus and de Kleer, 1988]. Dressler and Farquhar used a similar focusing mechanism in their model-based diagnostic system COCO to achieve efficient diagnosis of logic circuits [Dressler and Farquhar, 1990]. Such focusing techniques achieve pruning effects similar to our use of heuristic beam search in the AAA algorithm. Focusing eliminates environments that are not implied by some focus environments, while our heuristic beam search eliminates environments based on their evaluation metric.

Poole's implemented system, Theorist, is a general default and abductive reasoning system [Poole, 1989a]. Compared to ACCEL, Theorist also deals with default reasoning, and it handles full first-order predicate logic. However, the hypotheses (i.e., assumptions) that Theorist can make must be given to the system *a priori*, while all atoms are assumable in ACCEL (in the plan recognition domain). In addition, Theorist is not concerned with efficient inference and does not use caching to avoid redundant work, nor has it been tested on large problems.

7.2 Plan Recognition and Natural Language Understanding

7.2.1 Abductive Approaches

Charniak and McDermott were the first to formalize the inference of characters' plans and goals from their states and actions as abduction [Charniak and McDermott, 1985]. That is, understanding is the search for the best explanation of the causal reasons behind characters' behavior. This allows us to study such inferences in the larger context of inferring cause from effect.

Several research efforts have since adopted an abductive approach to text understanding. In [Charniak, 1988], it is shown that noun-phrase reference determination can be achieved by an abductive unification procedure that allows for unifying two entities if it is consistent to do so. This is equivalent to making the abductive assumption that the two entities are equal in an abductive proof of the input sentences.

Hobbs *et. al.* have used abduction to solve the four local pragmatics problems of text understanding: reference resolution, compound nominal interpretation, syntactic ambiguity resolution, and metonymy resolution [Hobbs *et al.*, 1988]. It is claimed that the abductive approach has "resulted in a dramatic simplification of how the problem of interpreting texts is conceptualized", and that "it also suggests an elegant and thorough integration of syntax, semantics, and pragmatics", by combining the idea of interpretation as abduction and that of parsing as deduction.

The work reported here differs from those of [Charniak, 1988] and [Hobbs *et al.*, 1988] in that unlike their emphasis on mostly linguistic issues like noun-phrase reference determination and syntactic ambiguity resolution, ACCEL is concerned with recognizing characters' plans in a narrative text. The work of [Charniak, 1986] also dealt with plan recognition, but evaluated explanations based on their simplicity, as opposed to our coherence metric. In addition, unlike their use of marker passing to restrict the search for explanations, we used a form of beam search to efficiently construct explanations.

7.2.2 Probabilistic Approaches

Inferring cause from effect is an inherently uncertain process — it is only plausible inference. There are usually many alternative causes for some observed effect given the relevant background knowledge. Statistics and probability theory is the established discipline of study that deals with uncertainty. As such, it comes as no surprise that much work in medical diagnosis is concerned with uncertain or probabilistic reasoning. Within AI, Pearl has done extensive work on probabilistic reasoning [Pearl, 1988].

Resolving ambiguity in natural language understanding can also be formulated as reasoning under uncertainty. Recently, Charniak and Goldman have adopted a probabilistic approach to text understanding [Charniak and Goldman, 1989; Charniak and Goldman, 1991; Goldman, 1990]. In this approach, a Bayesian network is constructed from the input sentences. Based on knowledge about the various prior and posterior probabilities of the nodes in a network, and making some appropriate assumptions about conditional independence, the conditional probabilities of various abduced events given the observations stated in the text are computed. The abduced event with the highest conditional probability constitutes the preferred interpretation of the text. Engineering the numerous prior and posterior probabilities of the nodes in a network is a weakness of this approach. Furthermore, as we have demonstrated in Chapter 3, the interpretation selected by this approach depends quite critically on the specific values assigned to the various probabilities, and that reasonable probability values may result in the wrong interpretation being selected. Also, selecting interpretations based solely on probability fails to capture the importance of text coherence.

7.2.3 Non-Abductive Approaches

Two early approaches to narrative understanding are script-based and plan-based understanding. In the script-based approach used by SAM [Cullingford, 1978], knowledge of stereotypical events are used to guide the understanding process. In the plan-based approach used by PAM [Wilensky, 1978; Wilensky, 1983], knowledge about the actions, plans and goals of characters are used to connect the observed states and actions to their high level plans and goals. The realization that a complete understanding of narratives requires knowledge of events, plans and goals characterizes these early approaches [Schank and Abelson, 1977].

Granger's program also makes inferences about characters' goals to understand stories [Granger, 1980a]. His program prefers the most *parsimonious* interpretation, defined as one in which "the fewest number of inferred goals of a story character account for the maximum number of his actions". As such, his preference criterion is simplicity as opposed to the coherence metric used in ACCEL.

Alterman proposed a theory of seven coherence relations to relate the events and states in a narrative [Alterman, 1985]. In his research, understanding a narrative involves finding the coherence relations between the various concepts in the text. This is accomplished by constructing a dictionary of concepts related by the seven coherence relations, matching text against this dictionary, and using the organization of the concepts in the dictionary to organize the instances of concepts that appear in the text.

The research of Norvig involves the use of marker passing mechanism to make inferences from narratives [Norvig, 1987]. The knowledge base is structured in a semantic network where a link between two nodes can be one of several types. A predetermined, fixed set of inference paths specified as regular expressions determine the valid path links that a marker can travel. Collision of markers at some intermediate node constitutes a possible inference. Competing inference paths are then evaluated by a separate evaluator. The weakness of this approach is that in a sizable knowledge base, the spreading of markers can still lead to many possible path collisions even when constrained by the predetermined set of allowable inference paths. Furthermore, the semantics of these predetermined regular-expression-style inference paths is unclear and the paths appear to be created solely for the convenience of constraining marker movement to make the inferences desired.

Previous research tends to ignore the fact that inferences drawn from text are only *plausible* inferences, and as such, they may subsequently turn out to be erroneous and need to be retracted. (An exception is the work of Granger and Eiselt who explicitly modeled such interpretation change [Granger, 1980a; Granger, 1980b; Eiselt, 1987].) Our abductive approach can handle interpretation change easily, as illustrated by the example in Section 3.5. Another issue in making plausible inferences from text is the selection of the preferred inferences given many alternatives. This is the notorious "frame selection" problem [Charniak, 1978] and has not been satisfactorily dealt with in previous research. Our work addresses this issue by efficiently constructing and evaluating alternative interpretations and selecting one based on explanatory coherence.

In the area of plan recognition for dialog understanding, Allen and Perrault were the first researchers to work on the inference of an agent's intentions from his utterances [Allen and Perrault, 1980]. Plan inference rules and heuristics were used to control the search for a plan. Their theory of plan recognition can also account for when to provide more information than explicitly requested, and how to interpret sentence fragments and indirect speech acts. However, their theory can only handle single utterances, and the plan inference rules make plausible deduction in an *ad hoc* way. Litman has extended this plan recognition model for dialog understanding to handle dialogs with sequences of utterances and embedded subdialogs [Litman and Allen, 1987]. She introduces the distinction between domain plans, which are the speaker plans that form the topic of conversation, and discourse plans, which are the speaker plans that relate utterances to the domain plans. Her model can account for subdialogs such as clarification, correction, and topic change. However, her approach (as well as that of Allen and Perrault) does not deal with the issue of interpretation change.

One shortcoming of these non-abductive approaches is that the underlying inference processes tend to be rather *ad hoc* and not based on any general, logical foundation. A logic-based approach offers an expressive representation language — first-order predicate calculus, with a clear, well-understood semantics. Although the inferences made in plan recognition are most emphatically not deductive, we believe that modeling them as inferring the assumptions sufficient to complete a deductive proof of the input observations is an elegant formalization. Abduction gives plan recognition inference a firm semantic footing. In addition, the inference process, although itself not deductive, depends on the familiar, well-understood notion of deduction.

7.3 Diagnosis

Reggia *et. al.* developed the generalized set covering (GSC) model [Reggia *et al.*, 1983; Reggia *et al.*, 1985] in which each disorder (disease) directly causes a set of manifestations (symptoms). The GSC model is essentially a propositional abduction model in which abductive proofs are restricted to be of depth one. Allemang *et. al.* have used a similar abduction model and an approximate algorithm to compute parsimonious diagnoses in a system that performs antibody identification in the domain of red blood cell typing [Allemang *et al.*, 1987]. ACCEL is more general in that it deals with first-order Horn clauses, and the explanations constructed can be of any depth.

Cox and Pietrzykowski have developed a general abductive inference procedure for computing fundamental causes of any observation stated as a first-order predicate calculus sentence [Cox and Pietrzykowski, 1986; Cox and Pietrzykowski, 1987]. A cause is any conjunct that logically entails the observation given the background knowledge, and a fundamental cause is a cause that is minimal (least general), acceptable (consistent), nontrivial (related to the knowledge base), and basic (not an intermediate cause). The minimality condition is a form of simplicity measure, i.e., do not assume anything more than what is sufficient to complete a proof of the observation. Their theory of abduction falls under the category of most specific abduction. However, their inference procedure does not utilize caching to improve efficiency, and it has been tested only on diagnostic problems in logic circuits.

Model-based diagnosis has recently been a very active area of research in AI [Reiter, 1987; de Kleer and Williams, 1987; de Kleer and Williams, 1989; Struss and Dressler, 1989; Hamscher, 1990]. In model-based diagnosis, an underlying model of a device's correct structure and behavior is given to a diagnostic system. Diagnosis proceeds from first principles using such a device model to explain discrepancies between its faulty and normal behavior.

Reiter proposed a formal theory of diagnosis from first principles that reasons from the system description and observations of the system behavior [Reiter, 1987]. In particular, an algorithm is presented that will compute all minimal diagnoses of a device, including multiple-fault diagnoses. A diagnosis in his theory is a set of normality and abnormality assumptions about the device components that are consistent with the observations and the system description. This theory provides the formal justification for the work of [Davis, 1984] and [de Kleer and Williams, 1987].

De Kleer and Williams implemented a diagnostic system, General Diagnostic Engine (GDE), which is capable of diagnosing failures due to multiple faults [de Kleer and Williams, 1987]. The system uses an ATMS to efficiently compute minimal diagnoses. It can also propose additional measurements to further localize faults. This is achieved via a method that minimizes the expected entropy of candidate probabilities. GDE has been applied to the diagnosis of logic circuits.

One shortcoming of GDE is that it does not have any knowledge of fault models, which are the common behavioral modes of a malfunctioned component. The lack of such knowledge may result in GDE proposing highly implausible diagnosis, like a light bulb is faulty in such a way that it is lit without any voltage supply. To overcome this problem, de Kleer and Williams subsequently built Sherlock which diagnosed a device based on knowledge about the normal behavior as well as common fault mode behavior of the components of a device [de Kleer and Williams, 1989]. The ability to diagnose unanticipated faults not known in advance is still retained.

The GDE+ system of Struss and Dressler has similar capability [Struss and Dressler, 1989].

Coco, the system of Dressler and Farquhar, also performs model-based diagnosis of logic circuits based on an ATMS [Dressler and Farquhar, 1990]. In order to focus diagnosis on the promising candidates, they used a focusing technique to restrict the propagation of environments in the ATMS. Empirical results indicate that significant improvement in diagnostic efficiency can be achieved using the focusing technique.

The work of Hamscher applied hierarchical decomposition of a device to better manage the computational complexity of computing diagnoses [Hamscher, 1990]. His system also incorporated fault models, and was applied to diagnosing digital circuits.

Much research in diagnosis has been applied to diagnosing digital circuits, which are representative only of physical devices with static, persistent states. Although there has been a great deal of research on modeling and simulating dynamic systems [Kuipers, 1986; Forbus, 1984], there have been few attempts to apply general, model-based diagnostic methods to them. Mimic, the system of [Dvorak and Kuipers, 1989] diagnoses continuous dynamic systems, but it assumes that all the fault models are pre-enumerated [Dvorak and Kuipers, 1989] (or that only single faults are present [Dvorak, 1992]). As such, unanticipated faults (or multiple, simultaneous, and interacting faults) will not be detected by the system.

The work of [Ng, 1991; Ng, 1990] attempts to address such deficiency by diagnosing dynamic systems using a general, model-based approach. It applies Reiter's algorithm incrementally to successive qualitative states of a dynamic system. The resulting algorithm, Inc-Diagnose, handles multiple faults correctly and does not require explicit fault models. An implicit assumption in Reiter's algorithm is that all observations are available at the start of diagnosis, which is often not the case. Since taking measurements is generally expensive, it is preferable to intermix measurement with fault hypothesis generation, especially for dynamic, continuous systems. Inc-Diagnose overcomes such limitations.

So far, the related research in model-based diagnosis cited above falls under the class of *consistency-based* diagnosis, in which a diagnosis is a set of normality and abnormality assumptions that are *consistent* with the observations and the system description. This is in contrast with the *abductive* approach to diagnosis used in Accel, in which normality and abnormality assumptions about the device components together with the system description must *imply* or *explain* the observations. Even though these two approaches may look quite different at first glance, the work of [Poole, 1988; Poole, 1989b; Konolige, 1992] has shown that under some restrictive conditions, the diagnoses computed by the consistency-based and abductive approaches are identical. Specifically, Poole showed that the two approaches are equivalent for propositional theories [Poole, 1988], and Konolige extended the conditions under which equivalence holds to general first-order causal theories allowing for correlations, uncertainty, and acyclicity in the causal structure. However, Konolige has reported that "the utility of the consistency based method is open to question", since in explanatory diagnostic tasks, "the answers it produces may have elements that are not relevant to a causal explanation" [Konolige, 1992, page 257].

7.4 Abduction in Other Domains

Thagard has independently proposed a computational theory of explanatory coherence and applied it to the evaluation of scientific theories [Thagard, 1989]. However, his theory of explanatory coherence consists of seven principles — symmetry, explanation, analogy, data priority, contradiction, acceptability, and system coherence. Independent criteria like simplicity and connectedness have been collapsed into one measure which he termed "explanatory coherence".

O'Rorke *et. al.* have modeled scientific theory formation as abduction [O'Rorke *et al.*, 1989]. They illustrated how some of Lavoisier's key insights during the Chemical Revolution can be viewed as examples of theory formation by abduction. It is suggested that "automated abduction is a key to advancing beyond the 'routine theory revision' methods developed in early AI research towards automated reasoning systems capable of 'world model revision' ". Their system differs from ACCEL in that it is a theory revision system designed to make changes to the rules in the underlying domain theory, while ACCEL assumes that its domain theory is correct.

Subramanian and Mooney have built a multistrategy learning system, BRACE, which combines abduction and theory revision to incorporate observations into a domain theory [Subramanian, 1992; Subramanian and Mooney, 1991]. BRACE is capable of changing both assumptions and rules in a domain theory to account for new observations, in contrast to ACCEL which does not modify the rules in the domain theory. When an explanation becomes inconsistent, BRACE will attempt to revise the assumptions to arrive at a consistent explanation, whereas ACCEL simply discards any inconsistent explanations.

Chapter 8

Future Work

Future research issues can be broadly classified into three areas: representation and algorithms, natural language understanding, and diagnosis.

8.1 Representation and Algorithms

The axioms allowed in ACCEL are restricted to first-order Horn-clause axioms for efficiency reasons, since linear resolution with Horn-clauses is in general more efficient than binary resolution with general clauses. However, the need for full first-order predicate logic representation, and hence, a general full first-order abduction algorithm may arise in the future. Stickel has already showed how his upside-down meta-interpretation method for abduction can be extended to deal with non-Horn clauses [Stickel, 1991]. The abduction algorithms of [Cox and Pietrzykowski, 1987; Poole, 1989a; Ginsberg, 1989] are also suitable for general first-order theories, although their algorithms do not involve caching.

In the current version of the AAA algorithm, consistency checking is accomplished by using a pre-determined list of nogoods and by procedural code. This is done purely for efficiency reasons. However, the current method for consistency checking is not as general. An earlier version of ACCEL [Ng and Mooney, 1991b; Ng and Mooney, 1991a] performed consistency checking by finding the logical consequences of the assumptions made in an explanation and the domain theory ($A \cup T$) via forward chaining on the Horn clause axioms in T (some of them are of the form $P_1 \wedge \ldots \wedge P_r \rightarrow false$) up to some preset depth limit. If $false$ is not derived within the depth limit, $A \cup T$ will be considered consistent.

The efficiency of ACCEL can be further improved by compiling the Horn-clauses in the knowledge base, in the same way that the efficiency of a deductive theorem prover can be greatly improved via clause compilation [Stickel, 1988b; Norvig, 1992].

One shortcoming of the AAA algorithm is that the queue of best partial explanations maintained may become empty at some point in computing the abductive proofs. This can occur if the beam width β_{intra} is not sufficiently large and all best partial explanations become inconsistent after adding a new input atom. A better approach would have the capability of recovering from an empty beam of explanations by reconciling the contradiction detected in an inconsistent explanation. Some assumption in a nogood (i.e., the set of assumptions that implies falsity) can be retracted and the remaining unexplained observations are proved via abduction again. Such contradiction resolution resembles belief revision for a justification-based truth maintenance system [Doyle, 1979]. The work of Subramanian addresses this issue [Subramanian, 1992].

8.2 Natural Language Understanding

ACCEL is currently only able to deal with the plan recognition aspect of text understanding. As mentioned in the related work section, abductive reasoning can also model noun-phrase reference determination [Charniak, 1988] and syntactic ambiguity resolution [Hobbs *et al.*, 1988]. ACCEL needs to be extended to include parsing of input sentences, and resolving lexical and syntactic ambiguity.

It is often the case that input atoms are too specific and cannot be directly deduced from abductive assumptions. This problem has been reported in [Charniak, 1987; Ng and Mooney, 1989]. For instance, in the sentences "John went to the supermarket. He bought some milk.", assuming that John was shopping at the supermarket only allows us to derive that he would buy some food, but not necessarily milk. This problem can be overcome by generalizing explanations to include abductive proofs of *logical consequences* of the input atoms. The difficulty in this generalized definition of abductive explanations is to determine what are the relevant and interesting consequences to derive and explain, since deriving all possible consequences is clearly intractable. An example of methods to control forward inferences is that developed for automated knowledge integration [Murray, 1988].

The plan recognition knowledge in ACCEL primarily encodes stereotypical knowledge as in a script-based system like SAM [Cullingford, 1978]. As such, ACCEL is not capable of handling novel plans, which involve actions not explicitly defined as part of a common plan, yet these actions are causally related and accomplish some high-level goal of an agent. In addition, ACCEL currently fails to handle common substeps that are shared by multiple plans, for instance, that a going action is part of both the supermarket-shopping and robbing plans. Overloading of actions to simultaneously achieve multiple goals is known to be a common occurrence [Pollack, 1989]. Furthermore, ACCEL is currently restricted to abducing high-level plans from observed actions, but not observed states. Hence, it will fail to abduce, for instance, that an agent has the restaurant-dining plan when told that he is hungry. We can of course write additional axioms to assert that a high-level plan implies an observed state, but to do so correctly, we must also assert when a state holds relative to the time of occurrence of plans and actions. For example, a person is only hungry before dining at a restaurant but not after. In other words, to properly handle the abduction of high-level plans from observed states, ACCEL must be able to reason more generally about time, and the preconditions and effects of actions in terms of the states that an action enables or disables. Extending ACCEL to correctly handle these deficiencies is an important area for future research.

Currently, explanations in the plan recognition domain are evaluated solely on coherence. Future work needs to integrate likelihood information as an part of the evaluation criterion, perhaps as a measure secondary to coherence.

8.3 Diagnosis

The work of [Poole, 1988; Poole, 1989b; Konolige, 1992] has revealed some interesting relationships between consistency-based and abductive diagnosis, which are two major paradigms in model-based diagnosis. To what extent do the two approaches coincide and differ, especially in practical terms such as ease of representation and diagnostic efficiency, remains to be investigated.

Our research does not focus on gathering additional measurements to further differentiate and narrow the diagnostic candidates. Intelligently selecting which component output and which additional sensor values to measure is important in achieving efficient diagnosis. For example, in GDE [de Kleer and Williams, 1987], additional measurements is gathered via a method that minimizes the expected entropy of candidate probabilities. Future work needs to extend ACCEL to incorporate intelligent experimentation.

The use of quantitative information in qualitative simulation can greatly reduce the ambiguity of the qualitative behavior of a dynamic system [Kuipers and Berleant, 1988]. The added precision allows better differential diagnosis [Dvorak, 1992]. The ability to monitor a dynamic system over time and performs diagnosis in real time as the device operates is also important [Dvorak, 1992]. Extending ACCEL to deal with quantitative knowledge and device monitoring are important future research issues.

As mentioned earlier, normality-based diagnosis is more flexible but may generate implausible diagnoses, while fault-based diagnosis requires explicit knowledge of fault models. To overcome the limitation of explicitly knowing all fault models in advance, we can instead develop a method to automatically acquire fault models over time. This can be accomplished by generalizing the common input-output behavior patterns as summarized by the parameterized abnormality assumptions of a component (e.g., $ab(X, U, V, W, T)$). Such a learning module would improve the diagnostic accuracy of ACCEL by recognizing the common fault modes of a device component.

Chapter 9

Conclusion

Finding explanations for observed phenomena underlies a diverse set of intelligent activities, ranging from natural language understanding, diagnosis, scientific theory formation, to image interpretation. The ubiquity of explanation underscores its importance as a research topic in artificial intelligence. In this thesis, we view explanation as logical abduction, which serves as a unifying formalism for explanation.

This thesis has made several important contributions:

1. We have demonstrated the practical feasibility of a general abductive approach to explanation by successfully building a domain-independent system, ACCEL, that is general enough to perform both plan recognition and diagnosis, yet efficient enough to be of practical utility. We support this claim by extensively evaluating the system on 50 narrative texts in the plan recognition domain, on 50 real patient cases in the set covering diagnosis domain, and on 10 model-based diagnosis scenarios each on an adder, a temperature controller, and the water balance system of the human kidney. Except for the adder circuit, each of the knowledge bases contains hundreds of Horn-clause rules.
2. We have developed a novel evaluation criterion, explanatory coherence, to evaluate the quality of explanations in the plan recognition domain. We present empirical results indicating that our coherence metric outperforms the simplicity metric in selecting the best explanation in the plan recognition domain. Our coherence-based approach performs as well as the probabilistic approach of plan recognition, but without the need to engineer numerous prior and posterior probabilities.
3. We present empirical evidence showing that caching of previously computed explanations is critical to the efficiency of an abduction algorithm. Specifically, speedup of more than an order of magnitude has been obtained on our test problems.

In summary, this thesis has demonstrated via an implemented system that general and efficient abduction for the tasks of plan recognition and diagnosis is indeed possible, and the future holds much promise for such a general abductive approach to explanation.

Appendix A

Plan Recognition

The knowledge bases and test data are listed in Lisp notation. The Lisp variables listed in the knowledge bases include:

1. *brules*: a list of rules in the domain theory;
2. *facts*: a list of facts in the domain theory;
3. *nogoods*: a list of nogoods;
4. *assumption-nogoods*: a list of "assumption-nogoods", as explained in subsection 3.4.2;
5. *inter-batch-beam-width*: β_{inter};
6. *intra-batch-beam-width*: β_{intra};
7. *bchain-depth*: the maximum depth of abductive explanations to pursue for input atoms;
8. *caching*: a flag to indicate whether caching should be used;
9. *factoring*: a flag to indicate whether factoring of assumptions should be performed;
10. *remove-superset?*: a flag to indicate whether supersets should be removed from the label of a subgoal;
11. *remove-superset-fn*: the function to use for removing supersets in the label of a subgoal;
12. *explanation-eval-metric*: the evaluation metric for explanations;
13. *compute-estimate-fn*: the function for estimating the quality of an individual environment while taking the cross product of labels;
14. *combine-estimates-fn*: the function for estimating the quality of a combined environment while taking the cross product of labels;
15. *predicate-specific-abduction*: a flag to indicate whether predicate specific abduction should be used;
16. *assumable-predicates*: a list of assumable predicates; and
17. *free-assumption-predicates*: a list of predicates that can be assumed without affecting the simplicity metric.

Note that in the appendices, variables in Horn-clauses are preceded by "?".

A.1 Knowledge Base

The knowledge base for the plan recognition domain is listed below.

```
(setf *brules* '(
; shopping

((inst ?g going) <- (inst ?s shopping) (go-step ?s ?g))
((goer ?g ?p) <- (inst ?s shopping) (go-step ?s ?g) (shopper ?s ?p))
((dest-go ?g ?str) <- (inst ?s shopping) (go-step ?s ?g) (store ?s ?str))
((inst ?sp shopping-place) <- (inst ?s shopping) (store ?s ?sp))

((inst ?f finding) <- (inst ?s shopping) (find-step ?s ?f))
((finder ?f ?a) <- (inst ?s shopping) (find-step ?s ?f) (shopper ?s ?a))
((thing-found ?f ?tf) <- (inst ?s shopping) (find-step ?s ?f)
                         (thing-shopped-for ?s ?tf))

((inst ?b buying) <- (inst ?s shopping) (buy-step ?s ?b))
((buyer ?b ?p) <- (inst ?s shopping) (buy-step ?s ?b) (shopper ?s ?p))
((thing-bought ?b ?tb) <- (inst ?s shopping) (buy-step ?s ?b)
                          (thing-shopped-for ?s ?tb))

((inst ?p paying) <- (inst ?b buying) (pay-step ?b ?p))
((payer ?p ?a) <- (inst ?b buying) (pay-step ?b ?p) (buyer ?b ?a))
((thing-paid ?p ?tp) <- (inst ?b buying) (pay-step ?b ?p)
                        (thing-bought ?b ?tp))

((inst ?str smarket) <- (inst ?s smarket-shopping) (store ?s ?str))
((inst ?f food) <- (inst ?s smarket-shopping) (thing-shopped-for ?s ?f))

((inst ?ls liquor-store) <- (inst ?s liqst-shopping) (store ?s ?ls))
((inst ?l liquor) <- (inst ?s liqst-shopping) (thing-shopped-for ?s ?l))

; robbing

((inst ?g getting) <- (inst ?r robbing) (get-weapon-step ?r ?g))
((agent-get ?g ?a) <- (inst ?r robbing) (get-weapon-step ?r ?g)
                      (robber ?r ?a))
((patient-get ?g ?w) <- (inst ?r robbing) (get-weapon-step ?r ?g)
                        (weapon-rob ?r ?w))

((inst ?g going) <- (inst ?r robbing) (go-step ?r ?g))
((goer ?g ?a) <- (inst ?r robbing) (go-step ?r ?g) (robber ?r ?a))
((dest-go ?g ?p) <- (inst ?r robbing) (go-step ?r ?g) (place-rob ?r ?p))

((inst ?p pointing) <- (inst ?r robbing) (point-weapon-step ?r ?p))
((agent-point ?p ?a) <- (inst ?r robbing) (point-weapon-step ?r ?p)
                        (robber ?r ?a))
((patient-point ?p ?a) <- (inst ?r robbing) (point-weapon-step ?r ?p)
                          (victim-rob ?r ?a))
((instr-point ?p ?i) <- (inst ?r robbing) (point-weapon-step ?r ?p)
                        (weapon-rob ?r ?i))
((inst ?i weapon) <- (inst ?r robbing) (weapon-rob ?r ?i))
```

```
((inst ?g getting) <- (inst ?r robbing) (get-valuable-step ?r ?g))
((agent-get ?g ?a) <- (inst ?r robbing) (get-valuable-step ?r ?g)
                       (robber ?r ?a))
((patient-get ?g ?t) <- (inst ?r robbing) (get-valuable-step ?r ?g)
                         (thing-robbed ?r ?t))
((from-get ?g ?a) <- (inst ?r robbing) (get-valuable-step ?r ?g)
                      (victim-rob ?r ?a))
((inst ?t valuable) <- (inst ?r robbing) (thing-robbed ?r ?t))

; restaurant dining

((inst ?g going) <- (inst ?d rest-dining) (go-step ?d ?g))
((goer ?g ?a) <- (inst ?d rest-dining) (go-step ?d ?g) (diner ?d ?a))
((dest-go ?g ?r) <- (inst ?d rest-dining) (go-step ?d ?g)
                    (restaurant ?d ?r))
((inst ?r restaurant) <- (inst ?d rest-dining) (restaurant ?d ?r))

((inst ?o ordering) <- (inst ?d rest-dining) (order-step ?d ?o))
((agent-order ?o ?a) <- (inst ?d rest-dining) (order-step ?d ?o)
                         (diner ?d ?a))
((patient-order ?o ?p) <- (inst ?d rest-dining) (order-step ?d ?o)
                            (rest-thing-ordered ?d ?p))

((inst ?o drinking) <- (inst ?d rest-dining) (drink-step ?d ?o))
((drinker ?o ?a) <- (inst ?d rest-dining) (drink-step ?d ?o) (diner ?d ?a))
((patient-drink ?o ?p) <- (inst ?d rest-dining) (drink-step ?d ?o)
                            (rest-thing-drunk ?d ?p))
((instr-drink ?o ?p) <- (inst ?d rest-dining) (drink-step ?d ?o)
                          (rest-drink-straw ?d ?p))

((inst ?o paying) <- (inst ?d rest-dining) (pay-step ?d ?o))
((payer ?o ?a) <- (inst ?d rest-dining) (pay-step ?d ?o) (diner ?d ?a))
((thing-paid ?o ?p) <- (inst ?d rest-dining) (pay-step ?d ?o)
                         (rest-thing-ordered ?d ?p))

((inst ?g getting) <- (inst ?d drinking) (get-straw-step ?d ?g))
((agent-get ?g ?a) <- (inst ?d drinking) (get-straw-step ?d ?g)
                       (drinker ?d ?a))
((patient-get ?g ?p) <- (inst ?d drinking) (get-straw-step ?d ?g)
                         (instr-drink ?d ?p))

((inst ?p putting) <- (inst ?d drinking) (put-straw-step ?d ?p))
((agent-put ?p ?a) <- (inst ?d drinking) (put-straw-step ?d ?p)
                       (drinker ?d ?a))
((patient-put ?p ?a) <- (inst ?d drinking) (put-straw-step ?d ?p)
                         (instr-drink ?d ?a))
((inst ?a straw) <- (inst ?d drinking) (instr-drink ?d ?a))
((place-put ?p ?a) <- (inst ?d drinking) (put-straw-step ?d ?p)
                       (patient-drink ?d ?a))

((inst ?i ingesting) <- (inst ?d drinking) (ingest-step ?d ?i))
((agent-ingest ?i ?a) <- (inst ?d drinking) (ingest-step ?d ?i)
                          (drinker ?d ?a))
((patient-ingest ?i ?p) <- (inst ?d drinking) (ingest-step ?d ?i)
                         (patient-drink ?d ?p))
((instr-ingest ?i ?p) <- (inst ?d drinking) (ingest-step ?d ?i)
                         (instr-drink ?d ?p))

; going-by-vehicle

((inst ?g going) <- (inst ?v going-by-vehicle) (go-step ?v ?g))
((goer ?g ?a) <- (inst ?v going-by-vehicle) (go-step ?v ?g) (goer ?v ?a))
((dest-go ?g ?s) <- (inst ?v going-by-vehicle) (go-step ?v ?g)
                    (source-go ?v ?s))

((inst ?o getting-on) <- (inst ?v going-by-vehicle) (get-on-step ?v ?o))
((agent-get-on ?o ?a) <- (inst ?v going-by-vehicle) (get-on-step ?v ?o)
                         (goer ?v ?a))
((patient-get-on ?o ?w) <- (inst ?v going-by-vehicle) (get-on-step ?v ?o)
                           (vehicle ?v ?w))
((place-get-on ?o ?p) <- (inst ?v going-by-vehicle) (get-on-step ?v ?o)
                         (source-go ?v ?p))
((inst ?w vehicle) <- (inst ?v going-by-vehicle) (vehicle ?v ?w))

((inst ?s sitting) <- (inst ?v going-by-vehicle) (sit-step ?v ?s))
((agent-sit ?s ?a) <- (inst ?v going-by-vehicle) (sit-step ?v ?s)
                      (goer ?v ?a))
((patient-sit ?s ?p) <- (inst ?v going-by-vehicle) (sit-step ?v ?s)
                        (vehicle-seat ?v ?p))
((inst ?p seat) <- (inst ?v going-by-vehicle) (vehicle-seat ?v ?p))
((in ?p ?w) <- (inst ?v going-by-vehicle) (vehicle-seat ?v ?p)
               (vehicle ?v ?w))

((inst ?o getting-off) <- (inst ?v going-by-vehicle) (get-off-step ?v ?o))
((agent-get-off ?o ?a) <- (inst ?v going-by-vehicle) (get-off-step ?v ?o)
                          (goer ?v ?a))
((patient-get-off ?o ?w) <- (inst ?v going-by-vehicle) (get-off-step ?v ?o)
                            (vehicle ?v ?w))
((place-get-off ?o ?p) <- (inst ?v going-by-vehicle) (get-off-step ?v ?o)
                          (dest-go ?v ?p))

; going-by-bus, -taxi, -plane

((inst ?v bus) <- (inst ?b going-by-bus) (vehicle ?b ?v))
((inst ?v taxi) <- (inst ?t going-by-taxi) (vehicle ?t ?v))
((inst ?v plane) <- (inst ?p going-by-plane) (vehicle ?p ?v))

; going-by-bus

((inst ?s bus-station) <- (inst ?v going-by-bus) (source-go ?v ?s))

((inst ?g giving) <- (inst ?b going-by-bus) (give-token-step ?b ?g))
((giver ?g ?a) <- (inst ?b going-by-bus) (give-token-step ?b ?g)
                  (goer ?b ?a))
((recipient ?g ?a) <- (inst ?b going-by-bus) (give-token-step ?b ?g)
                      (bus-driver ?b ?a))
((occupation ?a busdriver) <- (inst ?b going-by-bus) (bus-driver ?b ?a))
((thing-given ?g ?t) <- (inst ?b going-by-bus) (give-token-step ?b ?g)
                          (token ?b ?t))
((inst ?t token) <- (inst ?b going-by-bus) (token ?b ?t))

; going-by-taxi

((inst ?p paying) <- (inst ?b going-by-taxi) (pay-step ?b ?p))
((payer ?p ?a) <- (inst ?b going-by-taxi) (pay-step ?b ?p) (goer ?b ?a))
((payee ?p ?a) <- (inst ?b going-by-taxi) (pay-step ?b ?p)
                  (taxi-driver ?b ?a))
((occupation ?a taxidriver) <- (inst ?b going-by-taxi)
                               (taxi-driver ?b ?a))

; going-by-plane

((inst ?s airport) <- (inst ?v going-by-plane) (source-go ?v ?s))

((inst ?s packing) <- (inst ?p going-by-plane) (pack-step ?p ?s))
((agent-pack ?s ?a) <- (inst ?p going-by-plane) (pack-step ?p ?s)
                       (goer ?p ?a))
((patient-pack ?s ?l) <- (inst ?p going-by-plane) (pack-step ?p ?s)
                         (plane-luggage ?p ?l))
((inst ?l bag) <- (inst ?p going-by-plane) (plane-luggage ?p ?l))

((inst ?b buying) <- (inst ?s going-by-plane) (buy-ticket-step ?s ?b))
((buyer ?b ?a) <- (inst ?s going-by-plane) (buy-ticket-step ?s ?b)
                  (goer ?s ?a))
((thing-bought ?b ?t) <- (inst ?s going-by-plane) (buy-ticket-step ?s ?b)
                         (plane-ticket ?s ?t))
((inst ?t ticket) <- (inst ?s going-by-plane) (plane-ticket ?s ?t))

; jogging

((inst ?d drinking) <- (inst ?j jogging) (drink-step ?j ?d))
((drinker ?d ?a) <- (inst ?j jogging) (drink-step ?j ?d) (jogger ?j ?a))
((patient-drink ?d ?a) <- (inst ?j jogging) (drink-step ?j ?d)
                          (jog-thing-drunk ?j ?a))
((instr-drink ?d ?a) <- (inst ?j jogging) (drink-step ?j ?d)
                        (jog-drink-straw ?j ?a))

; partying

((inst ?d drinking) <- (inst ?p partying) (drink-step ?p ?d))
((drinker ?d ?a) <- (inst ?p partying) (drink-step ?p ?d)
                    (agent-party ?p ?a))
((patient-drink ?d ?a) <- (inst ?p partying) (drink-step ?p ?d)
                          (party-thing-drunk ?p ?a))
((instr-drink ?d ?a) <- (inst ?p partying) (drink-step ?p ?d)
                        (party-drink-straw ?p ?a))
))

(setf *facts* nil)

(setf *nogoods* nil)
```

```
(setf *assumption-nogoods* '(
((go-step ?s ?g) (goer ?g ?p))
((go-step ?s ?g) (dest-go ?g ?str))
((find-step ?s ?f) (finder ?f ?a))
((find-step ?s ?f) (thing-found ?f ?tf))
((buy-step ?s ?b) (buyer ?b ?p))
((buy-step ?s ?b) (thing-bought ?b ?tb))
((pay-step ?b ?p) (payer ?p ?a))
((pay-step ?b ?p) (payee ?p ?a))
((pay-step ?b ?p) (thing-paid ?p ?tp))
((get-weapon-step ?r ?g) (agent-get ?g ?a))
((get-weapon-step ?r ?g) (patient-get ?g ?w))
((point-weapon-step ?r ?p) (agent-point ?p ?a))
((point-weapon-step ?r ?p) (patient-point ?p ?a))
((point-weapon-step ?r ?p) (instr-point ?p ?i))
((get-valuable-step ?r ?g) (agent-get ?g ?a))
((get-valuable-step ?r ?g) (patient-get ?g ?t))
((get-valuable-step ?r ?g) (from-get ?g ?a))
((order-step ?d ?o) (agent-order ?o ?a))
((order-step ?d ?o) (patient-order ?o ?p))
((drink-step ?d ?o) (drinker ?o ?a))
((drink-step ?d ?o) (patient-drink ?o ?p))
((drink-step ?d ?o) (instr-drink ?o ?p))
((get-straw-step ?d ?g) (agent-get ?g ?a))
((get-straw-step ?d ?g) (patient-get ?g ?p))
((put-straw-step ?d ?p) (agent-put ?p ?a))
((put-straw-step ?d ?p) (patient-put ?p ?a))
((put-straw-step ?d ?p) (place-put ?p ?a))
((ingest-step ?d ?i) (agent-ingest ?i ?a))
((ingest-step ?d ?i) (patient-ingest ?i ?p))
((ingest-step ?d ?i) (instr-ingest ?i ?p))
((get-on-step ?v ?o) (agent-get-on ?o ?a))
((get-on-step ?v ?o) (patient-get-on ?o ?w))
((get-on-step ?v ?o) (place-get-on ?o ?p))
((sit-step ?v ?s) (agent-sit ?s ?a))
((sit-step ?v ?s) (patient-sit ?s ?p))
((get-off-step ?v ?o) (agent-get-off ?o ?a))
((get-off-step ?v ?o) (patient-get-off ?o ?w))
((get-off-step ?v ?o) (place-get-off ?o ?p))
((give-token-step ?b ?g) (giver ?g ?a))
((give-token-step ?b ?g) (recipient ?g ?a))
((give-token-step ?b ?g) (thing-given ?g ?t))
((pack-step ?p ?s) (agent-pack ?s ?a))
((pack-step ?p ?s) (patient-pack ?s ?l))
((buy-ticket-step ?s ?b) (buyer ?b ?a))
((buy-ticket-step ?s ?b) (thing-bought ?b ?t))
))

(setf *sort-hierarchy* '(
(any . (physical action))
(physical . (apparel bag food gift liquor place
            seat shelf straw ticket token valuable vehicle weapon))
(action . (buying courting drinking finding getting getting-off getting-on
          giving going ingesting jogging ordering packing partying paying
          pointing putting rest-dining robbing shopping sitting working))
(apparel . (shirt skirt trousers uniform))
(bag . (suitcase))
(food . (bread milk milkshake))
(gift . (flower jewelry))
(liquor . (bourbon))
(place . (airport bus-station park prison
         restaurant school shopping-place))
(valuable . (money))
(vehicle . (bus plane taxi))
(weapon . (gun knife))
(going . (going-by-vehicle))
(shopping . (liqst-shopping smarket-shopping))
(shopping-place . (liquor-store smarket))
(going-by-vehicle . (going-by-bus going-by-plane going-by-taxi))
))

(setf *inter-batch-beam-width* 10)
(setf *intra-batch-beam-width* 30)
(setf *bchain-depth* 3)
(setf *caching* t)
(setf *factoring* t)
(setf *remove-superset?* t)
(setf *remove-superset-fn* #'alphabetic-variant-subset?)
(setf *explanation-eval-metric* #'coherence-then-simplicity)
(setf *compute-estimate-fn* #'pr-compute-estimate-coherence)
(setf *combine-estimates-fn* #'pr-combine-estimates-coherence)

(setf *predicate-specific-abduction* nil)
(setf *free-assumption-predicates* nil)

(setf *unique-slot-value-predicates*
      '(go-step goer dest-go source-go vehicle
        shopper store thing-shopped-for
        find-step finder thing-found
        buy-step buyer thing-bought
        pay-step payer thing-paid
        get-straw-step get-weapon-step get-valuable-step
        agent-get patient-get from-get
        robber weapon-rob place-rob victim-rob thing-robbed
        point-weapon-step agent-point patient-point instr-point
        diner restaurant rest-thing-ordered
        rest-thing-drunk rest-drink-straw
        order-step agent-order patient-order
        drink-step drinker patient-drink instr-drink
        jogger jog-thing-drunk jog-drink-straw
        agent-party party-thing-drunk party-drink-straw
        put-straw-step agent-put patient-put place-put
        ingest-step agent-ingest patient-ingest instr-ingest
        get-on-step agent-get-on patient-get-on place-get-on
        sit-step agent-sit patient-sit vehicle-seat
        get-off-step agent-get-off patient-get-off place-get-off
        give-token-step giver recipient
        occupation thing-given token
        taxi-driver bus-driver
        pack-step agent-pack patient-pack plane-luggage
        buy-ticket-step plane-ticket))

(setf *plan-steps*
     '((shopping . (go-step find-step buy-step))
       (smarket-shopping . (go-step find-step buy-step))
       (liqst-shopping . (go-step find-step buy-step))
       (buying . (pay-step))
       (robbing . (get-weapon-step go-step point-weapon-step
                   get-valuable-step))
       (rest-dining . (go-step order-step drink-step pay-step))
       (drinking . (get-straw-step put-straw-step ingest-step))
       (going-by-vehicle . (go-step get-on-step sit-step get-off-step))
       (going-by-bus . (go-step give-token-step get-on-step
                        sit-step get-off-step))
       (going-by-taxi . (go-step get-on-step sit-step
                         pay-step get-off-step))
       (going-by-plane . (pack-step go-step buy-ticket-step get-on-step
                          sit-step get-off-step))
       (jogging . (drink-step))
       (partying . (drink-step))))
```

A.2 Training Stories

The 25 training stories are listed below. For each training story, we give: (1) the natural language sentences; (2) the input atoms representing the sentences; and (3) the correct assumptions that ACCEL should abduce.

```
1. "Jack went to the supermarket.
    He found some milk on the shelf.
    He paid for it."

((inst go1 going) (goer go1 jack1) (name jack1 jack) (dest-go go1 sm1)
 (inst sm1 smarket) (precede go1 find1) (inst find1 finding)
 (finder find1 jack1) (thing-found find1 milk1) (inst milk1 milk)
 (on milk1 shf1) (inst shf1 shelf) (precede find1 pay1)
 (inst pay1 paying) (payer pay1 jack1) (thing-paid pay1 milk1))

((inst ?s smarket-shopping) (go-step ?s go1) (find-step ?s find1)
 (buy-step ?s ?b) (pay-step ?b pay1) (shopper ?s jack1)
 (store ?s sm1) (thing-shopped-for ?s milk1) (name jack1 jack)
 (inst milk1 milk) (on milk1 shf1) (inst shf1 shelf)
 (precede go1 find1) (precede find1 pay1))

2. "Bill went to the supermarket.
    He paid for some milk."

((inst go2 going) (goer go2 bill2) (name bill2 bill) (dest-go go2 sm2)
 (inst sm2 smarket) (precede go2 pay2) (inst pay2 paying)
 (payer pay2 bill2) (thing-paid pay2 milk2) (inst milk2 milk))
```

```
((inst ?s smarket-shopping) (go-step ?s go2) (buy-step ?s ?b)
 (pay-step ?b pay2) (shopper ?s bill2) (store ?s sm2)
 (thing-shopped-for ?s milk2) (name bill2 bill) (inst milk2 milk)
 (precede go2 pay2))
```

3. "Jack gave the busdriver a token.
   He got off at the supermarket."

```
((inst give3 giving) (giver give3 jack3) (name jack3 jack)
 (recipient give3 bd3) (occupation bd3 busdriver)
 (thing-given give3 tk3) (inst tk3 token) (precede give3 getoff3)
 (inst getoff3 getting-off) (agent-get-off getoff3 jack3)
 (place-get-off getoff3 sm3) (inst sm3 smarket))

((inst ?s smarket-shopping) (inst ?b going-by-bus) (go-step ?s ?b)
 (give-token-step ?b give3) (get-off-step ?b getoff3) (shopper ?s jack3)
 (store ?s sm3) (bus-driver ?b bd3) (token ?b tk3) (name jack3 jack)
 (precede give3 getoff3))
```

4. "Jack got off the bus at the liquor-store.
   He pointed a gun at the owner."

```
((inst getoff4 getting-off) (agent-get-off getoff4 jack4) (name jack4 jack)
 (patient-get-off getoff4 bus4) (inst bus4 bus) (place-get-off getoff4 ls4)
 (inst ls4 liquor-store) (precede getoff4 point4) (inst point4 pointing)
 (agent-point point4 jack4) (instr-point point4 gun4) (inst gun4 gun)
 (patient-point point4 o4) (own o4 ls4))

((inst ?r robbing) (inst ?b going-by-bus) (go-step ?r ?b)
 (get-off-step ?b getoff4) (point-weapon-step ?r point4) (vehicle ?b bus4)
 (robber ?r jack4) (weapon-rob ?r gun4) (victim-rob ?r o4)
 (place-rob ?r ls4) (name jack4 jack) (inst ls4 liquor-store)
 (inst gun4 gun) (own o4 ls4) (precede getoff4 point4))
```

5. "Jack went to the liquor-store.
   He found some bourbon on the shelf."

```
((inst go5 going) (goer go5 jack5) (name jack5 jack)
 (dest-go go5 ls5) (inst ls5 liquor-store) (precede go5 find5)
 (inst find5 finding) (finder find5 jack5) (thing-found find5 bourbon5)
 (inst bourbon5 bourbon) (on bourbon5 shf5) (inst shf5 shelf))

((inst ?s liqst-shopping) (go-step ?s go5) (find-step ?s find5)
 (shopper ?s jack5) (store ?s ls5) (thing-shopped-for ?s bourbon5)
 (name jack5 jack) (inst bourbon5 bourbon) (on bourbon5 shf5)
 (inst shf5 shelf) (precede go5 find5))
```

6. "Bill went to the liquor-store.
   He pointed a gun at the owner."

```
((inst go6 going) (goer go6 bill6) (name bill6 bill) (dest-go go6 ls6)
 (inst ls6 liquor-store) (precede go6 point6) (inst point6 pointing)
 (agent-point point6 bill6) (instr-point point6 gun6)
 (inst gun6 gun) (patient-point point6 o6) (own o6 ls6))
```

```
((inst ?r robbing) (go-step ?r go6) (point-weapon-step ?r point6)
 (robber ?r bill6) (place-rob ?r ls6) (weapon-rob ?r gun6)
 (victim-rob ?r o6) (name bill6 bill) (inst ls6 liquor-store)
 (inst gun6 gun) (own o6 ls6) (precede go6 point6))
```

7. "Bill gave the busdriver a token."

```
((inst give7 giving) (giver give7 bill7) (name bill7 bill)
 (recipient give7 bd7) (occupation bd7 busdriver)
 (thing-given give7 tk7) (inst tk7 token))

((inst ?b going-by-bus) (give-token-step ?b give7) (goer ?b bill7)
 (bus-driver ?b bd7) (token ?b tk7) (name bill7 bill))
```

8. "Fred robbed the liquor-store.
    Fred pointed a gun at the owner."

```
((inst rob8 robbing) (robber rob8 fred8) (name fred8 fred)
 (place-rob rob8 ls8) (inst ls8 liquor-store) (inst point8 pointing)
 (agent-point point8 fred8) (instr-point point8 gun8)
 (inst gun8 gun) (patient-point point8 o8) (own o8 ls8))

((inst rob8 robbing) (point-weapon-step rob8 point8) (robber rob8 fred8)
 (weapon-rob rob8 gun8) (victim-rob rob8 o8) (place-rob rob8 ls8)
 (name fred8 fred) (inst ls8 liquor-store) (inst gun8 gun) (own o8 ls8))
```

9. "Bill got a gun.
    He went to the supermarket."

```
((inst get9 getting) (agent-get get9 bill9) (name bill9 bill)
 (patient-get get9 gun9) (inst gun9 gun) (precede get9 go9)
 (inst go9 going) (goer go9 bill9) (dest-go go9 sm9) (inst sm9 smarket))

((inst ?r robbing) (get-weapon-step ?r get9) (go-step ?r go9)
 (robber ?r bill9) (weapon-rob ?r gun9) (place-rob ?r sm9)
 (name bill9 bill) (inst gun9 gun) (inst sm9 smarket) (precede get9 go9))
```

10. "Fred went to the supermarket.
     He pointed a gun at the owner.
     He packed his bag.
     He went to the airport."

```
((inst go10 going) (goer go10 fred10) (name fred10 fred)
 (dest-go go10 sm10) (inst sm10 smarket) (precede go10 point10)
 (inst point10 pointing) (agent-point point10 fred10)
 (instr-point point10 gun10) (inst gun10 gun) (patient-point point10 o10)
 (own o10 sm10) (precede point10 pack10) (inst pack10 packing)
 (agent-pack pack10 fred10) (patient-pack pack10 bag10) (inst bag10 bag)
 (precede pack10 go10b) (inst go10b going) (goer go10b fred10)
 (dest-go go10b airport10) (inst airport10 airport))

((inst ?r robbing) (go-step ?r go10) (point-weapon-step ?r point10)
 (robber ?r fred10) (place-rob ?r sm10) (weapon-rob ?r gun10)
 (victim-rob ?r o10) (inst ?p going-by-plane) (pack-step ?p pack10)
 (go-step ?p go10b) (goer ?p fred10) (plane-luggage ?p bag10)
 (source-go ?p airport10) (name fred10 fred) (inst sm10 smarket)
 (inst gun10 gun) (own o10 sm10) (precede go10 point10)
 (precede point10 pack10) (precede pack10 go10b))

11. "Jack took the bus to the airport.
     He bought a ticket."

((inst go11 going) (goer go11 jack11) (name jack11 jack)
 (vehicle go11 bus11) (inst bus11 bus) (dest-go go11 airport11)
 (inst airport11 airport) (precede go11 buy11) (inst buy11 buying)
 (buyer buy11 jack11) (thing-bought buy11 tk11) (inst tk11 ticket))

((inst ?p going-by-plane) (inst go11 going-by-bus) (go-step ?p go11)
 (buy-ticket-step ?p buy11) (vehicle go11 bus11) (goer ?p jack11)
 (source-go ?p airport11) (plane-ticket ?p tk11) (name jack11 jack)
 (precede go11 buy11))

12. "Bill packed a suitcase.
     He went to the airport."

((inst pack12 packing) (agent-pack pack12 bill12) (name bill12 bill)
 (patient-pack pack12 sc12) (inst sc12 suitcase) (precede pack12 go12)
 (inst go12 going) (goer go12 bill12) (dest-go go12 airport12)
 (inst airport12 airport))

((inst ?p going-by-plane) (pack-step ?p pack12) (go-step ?p go12)
 (goer ?p bill12) (plane-luggage ?p sc12) (source-go ?p airport12)
 (name bill12 bill) (inst sc12 suitcase) (precede pack12 go12))

13. "Jack got on a bus.
     He got off at the park.
     Jack went to the supermarket."

((inst geton13 getting-on) (agent-get-on geton13 jack13)
 (name jack13 jack) (patient-get-on geton13 bus13) (inst bus13 bus)
 (precede geton13 getoff13) (inst getoff13 getting-off)
 (agent-get-off getoff13 jack13) (place-get-off getoff13 park13)
 (inst park13 park) (precede getoff13 go13) (inst go13 going)
 (goer go13 jack13) (dest-go go13 sm13) (inst sm13 smarket))

((inst ?b going-by-bus) (get-on-step ?b geton13) (get-off-step ?b getoff13)
 (goer ?b jack13) (vehicle ?b bus13) (dest-go ?b park13)
 (inst ?s smarket-shopping) (go-step ?s go13) (shopper ?s jack13)
 (store ?s sm13) (name jack13 jack) (inst park13 park)
 (precede geton13 getoff13) (precede getoff13 go13))

14. "Jack gave the busdriver a token.
     He got off at the park.
     He went to the airport.
     He got on a plane."

((inst give14 giving) (giver give14 jack14) (name jack14 jack)
 (recipient give14 bd14) (occupation bd14 busdriver)
 (thing-given give14 tk14) (inst tk14 token) (precede give14 getoff14)
 (inst getoff14 getting-off) (agent-get-off getoff14 jack14)
 (place-get-off getoff14 park14) (inst park14 park)
 (precede getoff14 go14) (inst go14 going) (goer go14 jack14)
 (dest-go go14 airport14) (inst airport14 airport) (precede go14 geton14)
 (inst geton14 getting-on) (agent-get-on geton14 jack14)
 (patient-get-on geton14 plane14) (inst plane14 plane))

((inst ?b going-by-bus) (give-token-step ?b give14)
 (get-off-step ?b getoff14) (goer ?b jack14) (bus-driver ?b bd14)
 (token ?b tk14) (dest-go ?b park14) (inst ?p going-by-plane)
 (go-step ?p go14) (get-on-step ?p geton14) (goer ?p jack14)
 (source-go ?p airport14) (vehicle ?p plane14) (name jack14 jack)
 (inst park14 park) (precede give14 getoff14) (precede getoff14 go14)
 (precede go14 geton14))

15. "Fred sat down on the bus.
     He went to the supermarket."

((inst sit15 sitting) (agent-sit sit15 fred15) (name fred15 fred)
 (patient-sit sit15 seat15) (inst seat15 seat) (in seat15 bus15)
 (inst bus15 bus) (precede sit15 go15) (inst go15 going)
 (goer go15 fred15) (dest-go go15 sm15) (inst sm15 smarket))

((inst ?s smarket-shopping) (go-step ?s go15) (shopper ?s fred15)
 (store ?s sm15) (inst go15 going-by-bus) (sit-step go15 sit15)
 (vehicle-seat go15 seat15) (vehicle go15 bus15) (name fred15 fred)
 (precede sit15 go15))

16. "Jack went to a restaurant.
     He got a milkshake."

((inst go16 going) (goer go16 jack16) (name jack16 jack)
 (dest-go go16 rest16) (inst rest16 restaurant) (precede go16 order16)
 (inst order16 ordering) (agent-order order16 jack16)
 (patient-order order16 ms16) (inst ms16 milkshake))

((inst ?d rest-dining) (go-step ?d go16) (diner ?d jack16)
 (restaurant ?d rest16) (order-step ?d order16) (rest-thing-ordered ?d ms16)
 (name jack16 jack) (inst ms16 milkshake) (precede go16 order16))

17. "Bill drank a milkshake with a straw."

((inst drink17 drinking) (drinker drink17 bill17) (name bill17 bill)
 (patient-drink drink17 ms17) (inst ms17 milkshake)
 (instr-drink drink17 str17) (inst str17 straw))

((inst drink17 drinking) (drinker drink17 bill17)
 (name bill17 bill) (patient-drink drink17 ms17)
 (inst ms17 milkshake) (instr-drink drink17 str17))

18. "Fred got off the bus at a restaurant.
     He got a milkshake."
```

```
((inst getoff18 getting-off) (agent-get-off getoff18 fred18)
 (name fred18 fred) (patient-get-off getoff18 bus18) (inst bus18 bus)
 (place-get-off getoff18 rest18) (inst rest18 restaurant)
 (precede getoff18 order18) (inst order18 ordering)
 (agent-order order18 fred18) (patient-order order18 ms18)
 (inst ms18 milkshake))

((inst ?d rest-dining) (go-step ?d ?b) (order-step ?d order18)
 (inst ?b going-by-bus) (get-off-step ?b getoff18) (vehicle ?b bus18)
 (diner ?d fred18) (restaurant ?d rest18) (rest-thing-ordered ?d ms18)
 (name fred18 fred) (inst ms18 milkshake) (precede getoff18 order18))

19. "Janet put a straw in a milkshake."

((inst put19 putting) (agent-put put19 janet19) (name janet19 janet)
 (patient-put put19 str19) (inst str19 straw) (place-put put19 ms19)
 (inst ms19 milkshake))

((inst ?d drinking) (put-straw-step ?d put19) (drinker ?d janet19)
 (instr-drink ?d str19) (patient-drink ?d ms19) (name janet19 janet)
 (inst ms19 milkshake))

20. "Bill got on a bus.
     He got off at a restaurant.
     He drank a milkshake with a straw."

((inst geton20 getting-on) (agent-get-on geton20 bill20)
 (name bill20 bill) (patient-get-on geton20 bus20) (inst bus20 bus)
 (precede geton20 getoff20) (inst getoff20 getting-off)
 (agent-get-off getoff20 bill20) (place-get-off getoff20 rest20)
 (inst rest20 restaurant) (precede getoff20 drink20) (inst drink20 drinking)
 (drinker drink20 bill20) (patient-drink drink20 ms20) (inst ms20 milkshake)
 (instr-drink drink20 str20) (inst str20 straw))

((inst ?d rest-dining) (go-step ?d ?b) (inst ?b going-by-bus)
 (get-on-step ?b geton20) (vehicle ?b bus20) (get-off-step ?b getoff20)
 (diner ?d bill20) (restaurant ?d rest20) (drink-step ?d drink20)
 (rest-thing-drunk ?d ms20) (rest-drink-straw ?d str20) (name bill20 bill)
 (inst ms20 milkshake) (precede geton20 getoff20)
 (precede getoff20 drink20))

21. "Bill took a bus to a restaurant.
     He drank a milkshake.
     He pointed a gun at the owner.
     He got some money from him."

((inst go21 going) (goer go21 bill21) (name bill21 bill)
 (vehicle go21 bus21) (inst bus21 bus) (dest-go go21 rest21)
 (inst rest21 restaurant) (precede go21 drink21) (inst drink21 drinking)
 (drinker drink21 bill21) (patient-drink drink21 ms21)
 (inst ms21 milkshake) (precede drink21 point21) (inst point21 pointing)
 (agent-point point21 bill21) (instr-point point21 gun21) (inst gun21 gun)
 (patient-point point21 o21) (own o21 rest21) (precede point21 get21)
 (inst get21 getting) (agent-get get21 bill21) (patient-get get21 money21)
 (inst money21 money) (from-get get21 o21))

((inst ?d rest-dining) (go-step ?d go21) (diner ?d bill21)
 (inst go21 going-by-bus) (restaurant ?d rest21) (drink-step ?d drink21)
 (rest-thing-drunk ?d ms21) (inst ?r robbing) (go-step ?r go21)
 (place-rob ?r rest21) (point-weapon-step ?r point21) (robber ?r bill21)
 (weapon-rob ?r gun21) (victim-rob ?r o21) (get-valuable-step ?r get21)
 (thing-robbed ?r money21) (name bill21 bill) (vehicle go21 bus21)
 (inst ms21 milkshake) (inst gun21 gun) (own o21 rest21)
 (inst money21 money) (precede go21 drink21) (precede drink21 point21)
 (precede point21 get21))

22. "Fred gave the busdriver a token.
    He got off the bus at the park.
    He went to a restaurant.
    He got some money from the owner."

((inst give22 giving) (giver give22 fred22) (name fred22 fred)
 (recipient give22 bd22) (occupation bd22 busdriver)
 (thing-given give22 tk22) (inst tk22 token) (precede give22 getoff22)
 (inst getoff22 getting-off) (agent-get-off getoff22 fred22)
 (patient-get-off getoff22 bus22) (inst bus22 bus)
 (place-get-off getoff22 park22) (inst park22 park) (precede getoff22 go22)
 (inst go22 going) (goer go22 fred22) (dest-go go22 rest22)
 (inst rest22 restaurant) (precede go22 get22) (inst get22 getting)
 (agent-get get22 fred22) (patient-get get22 money22)
 (inst money22 money) (from-get get22 o22) (own o22 rest22))

((inst ?b going-by-bus) (give-token-step ?b give22)
 (get-off-step ?b getoff22) (goer ?b fred22) (bus-driver ?b bd22)
 (token ?b tk22) (vehicle ?b bus22) (dest-go ?b park22)
 (inst ?r robbing) (go-step ?r go22) (get-valuable-step ?r get22)
 (robber ?r fred22) (place-rob ?r rest22) (thing-robbed ?r money22)
 (victim-rob ?r o22) (name fred22 fred) (inst park22 park)
 (inst rest22 restaurant) (inst money22 money) (own o22 rest22)
 (precede give22 getoff22) (precede getoff22 go22) (precede go22 get22))

23. "Jack took a taxi to the park."

((inst go23 going) (goer go23 jack23) (name jack23 jack)
 (vehicle go23 taxi23) (inst taxi23 taxi) (dest-go go23 park23)
 (inst park23 park))

((inst go23 going-by-taxi) (goer go23 jack23) (name jack23 jack)
 (vehicle go23 taxi23) (dest-go go23 park23) (inst park23 park))

24. "Bill took a taxi."

((inst go24 going) (goer go24 bill24) (name bill24 bill)
 (vehicle go24 taxi24) (inst taxi24 taxi))

((inst go24 going-by-taxi) (goer go24 bill24) (name bill24 bill)
 (vehicle go24 taxi24))
```

```
25. "Fred took a taxi to the bus-station.
     He got on a bus."

((inst go25 going) (goer go25 fred25) (name fred25 fred)
 (vehicle go25 taxi25) (inst taxi25 taxi) (dest-go go25 bus-station25)
 (inst bus-station25 bus-station) (precede go25 geton25)
 (inst geton25 getting-on) (agent-get-on geton25 fred25)
 (patient-get-on geton25 bus25) (inst bus25 bus))

((inst ?b going-by-bus) (go-step ?b go25) (inst go25 going-by-taxi)
 (goer ?b fred25) (source-go ?b bus-station25) (get-on-step ?b geton25)
 (vehicle ?b bus25) (name fred25 fred) (vehicle go25 taxi25)
 (precede go25 geton25))
```

A.3 Test Stories

```
1. "John found the bread.
    He paid for it."

((inst find1 finding) (finder find1 john1) (name john1 john)
 (thing-found find1 bread1) (inst bread1 bread) (precede find1 pay1)
 (inst pay1 paying) (payer pay1 john1) (thing-paid pay1 bread1))

((inst ?s shopping) (find-step ?s find1) (buy-step ?s ?b)
 (pay-step ?b pay1) (shopper ?s john1) (thing-shopped-for ?s bread1)
 (inst bread1 bread) (name john1 john) (precede find1 pay1))

2. "Bob got a gun.
    He got off the bus at the liquor store."

((inst get2 getting) (agent-get get2 bob2) (name bob2 bob)
 (patient-get get2 gun2) (inst gun2 gun) (precede get2 getoff2)
 (inst getoff2 getting-off) (agent-get-off getoff2 bob2)
 (patient-get-off getoff2 bus2) (inst bus2 bus)
 (place-get-off getoff2 ls2) (inst ls2 liquor-store))

((inst ?r robbing) (inst ?b going-by-bus) (get-weapon-step ?r get2)
 (go-step ?r ?b) (get-off-step ?b getoff2) (robber ?r bob2)
 (weapon-rob ?r gun2) (place-rob ?r ls2) (vehicle ?b bus2) (inst gun2 gun)
 (inst ls2 liquor-store) (name bob2 bob) (precede get2 getoff2))

3. "Bob went to the supermarket.
    He pointed a gun at the owner."

((inst go3 going) (goer go3 bob3) (name bob3 bob) (dest-go go3 sm3)
 (inst sm3 smarket) (precede go3 point3) (inst point3 pointing)
 (agent-point point3 bob3) (instr-point point3 gun3)
 (inst gun3 gun) (patient-point point3 o3) (own o3 sm3))

((inst ?r robbing) (go-step ?r go3) (point-weapon-step ?r point3)
 (robber ?r bob3) (place-rob ?r sm3) (weapon-rob ?r gun3)
 (victim-rob ?r o3) (inst sm3 smarket) (inst gun3 gun)
 (own o3 sm3) (name bob3 bob) (precede go3 point3))
```

4. "Bill packed a suitcase.
    He got off the bus at the airport."

```
((inst pack4 packing) (agent-pack pack4 bill4) (name bill4 bill)
 (patient-pack pack4 sc4) (inst sc4 suitcase) (precede pack4 getoff4)
 (inst getoff4 getting-off) (agent-get-off getoff4 bill4)
 (patient-get-off getoff4 bus4) (inst bus4 bus)
 (place-get-off getoff4 airport4) (inst airport4 airport))

((inst ?p going-by-plane) (inst ?b going-by-bus) (pack-step ?p pack4)
 (go-step ?p ?b) (get-off-step ?b getoff4) (goer ?p bill4)
 (plane-luggage ?p sc4) (vehicle ?b bus4) (source-go ?p airport4)
 (inst sc4 suitcase) (name bill4 bill) (precede pack4 getoff4))
```

5. "Jack gave the busdriver a token.
    He got off at the supermarket.
    He paid for some milk."

```
((inst give5 giving) (giver give5 jack5) (name jack5 jack)
 (recipient give5 bd5) (occupation bd5 busdriver) (thing-given give5 tk5)
 (inst tk5 token) (precede give5 getoff5) (inst getoff5 getting-off)
 (agent-get-off getoff5 jack5) (place-get-off getoff5 sm5)
 (inst sm5 smarket) (precede getoff5 pay5) (inst pay5 paying)
 (payer pay5 jack5) (thing-paid pay5 milk5) (inst milk5 milk))

((inst ?s smarket-shopping) (inst ?b going-by-bus) (go-step ?s ?b)
 (give-token-step ?b give5) (get-off-step ?b getoff5) (buy-step ?s ?u)
 (pay-step ?u pay5) (shopper ?s jack5) (bus-driver ?b bd5) (token ?b tk5)
 (store ?s sm5) (thing-shopped-for ?s milk5) (inst milk5 milk)
 (name jack5 jack) (precede give5 getoff5) (precede getoff5 pay5))
```

6. "Jack gave the busdriver a token.
    He got off at the restaurant.
    He paid for some milk."

```
((inst give6 giving) (giver give6 jack6) (name jack6 jack)
 (recipient give6 bd6) (occupation bd6 busdriver) (thing-given give6 tk6)
 (inst tk6 token) (precede give6 getoff6) (inst getoff6 getting-off)
 (agent-get-off getoff6 jack6) (place-get-off getoff6 rest6)
 (inst rest6 restaurant) (precede getoff6 pay6) (inst pay6 paying)
 (payer pay6 jack6) (thing-paid pay6 milk6) (inst milk6 milk))

((inst ?d rest-dining) (inst ?b going-by-bus) (go-step ?d ?b)
 (give-token-step ?b give6) (get-off-step ?b getoff6) (pay-step ?d pay6)
 (diner ?d jack6) (bus-driver ?b bd6) (token ?b tk6) (restaurant ?d rest6)
 (rest-thing-ordered ?d milk6) (inst milk6 milk) (name jack6 jack)
 (precede give6 getoff6) (precede getoff6 pay6))
```

7. "Fred got a gun.
    He got some money from the owner of the supermarket."

```
((inst get7 getting) (agent-get get7 fred7) (name fred7 fred)
 (patient-get get7 gun7) (inst gun7 gun) (precede get7 get7b)
 (inst get7b getting) (agent-get get7b fred7) (patient-get get7b money7)
 (inst money7 money) (from-get get7b o7) (own o7 sm7) (inst sm7 smarket))

((inst ?r robbing) (get-weapon-step ?r get7) (get-valuable-step ?r get7b)
 (robber ?r fred7) (weapon-rob ?r gun7) (thing-robbed ?r money7)
 (victim-rob ?r o7) (inst gun7 gun) (inst money7 money) (own o7 sm7)
 (inst sm7 smarket) (name fred7 fred) (precede get7 get7b))

8. "Fred got a gun.
   He went to the restaurant.
   He packed a suitcase."

((inst get8 getting) (agent-get get8 fred8) (name fred8 fred)
 (patient-get get8 gun8) (inst gun8 gun) (precede get8 go8)
 (inst go8 going) (goer go8 fred8) (dest-go go8 rest8)
 (inst rest8 restaurant) (precede go8 pack8) (inst pack8 packing)
 (agent-pack pack8 fred8) (patient-pack pack8 sc8) (inst sc8 suitcase))

((inst ?r robbing) (get-weapon-step ?r get8) (go-step ?r go8)
 (robber ?r fred8) (weapon-rob ?r gun8) (place-rob ?r rest8) (inst gun8 gun)
 (inst rest8 restaurant) (inst ?p going-by-plane) (pack-step ?p pack8)
 (goer ?p fred8) (plane-luggage ?p sc8) (inst sc8 suitcase)
 (name fred8 fred) (precede get8 go8) (precede go8 pack8))

9. "Tom went to the restuarant.
   He put a straw in a milkshake."

((inst go9 going) (goer go9 tom9) (name tom9 tom) (dest-go go9 rest9)
 (inst rest9 restaurant) (precede go9 put9) (inst put9 putting)
 (agent-put put9 tom9) (patient-put put9 str9) (inst str9 straw)
 (place-put put9 ms9) (inst ms9 milkshake))

((inst ?d rest-dining) (go-step ?d go9) (drink-step ?d ?dk)
 (put-straw-step ?dk put9) (diner ?d tom9) (restaurant ?d rest9)
 (rest-drink-straw ?d str9) (rest-thing-drunk ?d ms9)
 (inst ms9 milkshake) (name tom9 tom) (precede go9 put9))

10. "Bob took a taxi to the liquor store.
    He found some bourbon on the shelf."

((inst go10 going) (goer go10 bob10) (name bob10 bob)
 (vehicle go10 taxi10) (inst taxi10 taxi) (dest-go go10 ls10)
 (inst ls10 liquor-store) (precede go10 find10) (inst find10 finding)
 (finder find10 bob10) (thing-found find10 bourbon10)
 (inst bourbon10 bourbon) (on bourbon10 shf10) (inst shf10 shelf))

((inst ?s liqst-shopping) (inst go10 going-by-taxi) (go-step ?s go10)
 (find-step ?s find10) (shopper ?s bob10) (vehicle go10 taxi10)
 (store ?s ls10) (thing-shopped-for ?s bourbon10)
 (inst bourbon10 bourbon) (on bourbon10 shf10) (inst shf10 shelf)
 (name bob10 bob) (precede go10 find10))
```

```
11. "Jane took a taxi to the airport.
     She pointed a gun at the taxi-driver."

((inst go11 going) (goer go11 jane11) (name jane11 jane)
 (vehicle go11 taxi11) (inst taxi11 taxi) (dest-go go11 airport11)
 (inst airport11 airport) (precede go11 point11) (inst point11 pointing)
 (agent-point point11 jane11) (instr-point point11 gun11) (inst gun11 gun)
 (patient-point point11 td11) (occupation td11 taxidriver))

((inst ?p going-by-plane) (inst go11 going-by-taxi) (inst ?r robbing)
 (go-step ?p go11) (goer ?p jane11) (vehicle go11 taxi11)
 (source-go ?p airport11) (name jane11 jane) (point-weapon-step ?r point11)
 (robber ?r jane11) (weapon-rob ?r gun11) (victim-rob ?r td11)
 (inst gun11 gun) (taxi-driver go11 td11) (precede go11 point11))

12. "John got a gun.
     He took a taxi to the airport."

((inst get12 getting) (agent-get get12 john12) (name john12 john)
 (patient-get get12 gun12) (inst gun12 gun) (precede get12 go12)
 (inst go12 going) (goer go12 john12) (vehicle go12 taxi12)
 (inst taxi12 taxi) (dest-go go12 airport12) (inst airport12 airport))

((inst ?r robbing) (inst go12 going-by-taxi) (get-weapon-step ?r get12)
 (go-step ?r go12) (robber ?r john12) (weapon-rob ?r gun12)
 (vehicle go12 taxi12) (place-rob ?r airport12) (inst gun12 gun)
 (inst airport12 airport) (name john12 john) (precede get12 go12))

13. "Alice got a gun.
     She went to the bus-station."

((inst get13 getting) (agent-get get13 alice13) (name alice13 alice)
 (patient-get get13 gun13) (inst gun13 gun) (precede get13 go13)
 (inst go13 going) (goer go13 alice13) (dest-go go13 bus-station13)
 (inst bus-station13 bus-station))

((inst ?r robbing) (get-weapon-step ?r get13) (go-step ?r go13)
 (robber ?r alice13) (weapon-rob ?r gun13) (place-rob ?r bus-station13)
 (inst gun13 gun) (inst bus-station13 bus-station) (name alice13 alice)
 (precede get13 go13))

14. "John paid the taxi-driver.
     He got on the plane."

((inst pay14 paying) (payer pay14 john14) (name john14 john)
 (payee pay14 td14) (occupation td14 taxidriver) (precede pay14 geton14)
 (inst geton14 getting-on) (agent-get-on geton14 john14)
 (patient-get-on geton14 plane14) (inst plane14 plane))

((inst ?p going-by-plane) (inst ?t going-by-taxi) (go-step ?p ?t)
 (pay-step ?t pay14) (get-on-step ?p geton14) (goer ?p john14)
 (taxi-driver ?t td14) (vehicle ?p plane14) (name john14 john)
 (precede pay14 geton14))
```

15. "John took a cab to the airport."

```
((inst go15 going) (goer go15 john15) (name john15 john)
 (vehicle go15 taxi15) (inst taxi15 taxi) (dest-go go15 airport15)
 (inst airport15 airport))

((inst ?t going-by-plane) (inst go15 going-by-taxi) (go-step ?t go15)
 (goer ?t john15) (vehicle go15 taxi15) (source-go ?t airport15)
 (name john15 john))
```

16. "Fred got off the bus at the supermarket."

```
((inst getoff16 getting-off) (agent-get-off getoff16 fred16)
 (name fred16 fred) (patient-get-off getoff16 bus16) (inst bus16 bus)
 (place-get-off getoff16 sm16) (inst sm16 smarket))

((inst ?s smarket-shopping) (inst ?b going-by-bus) (go-step ?s ?b)
 (get-off-step ?b getoff16) (shopper ?s fred16) (vehicle ?b bus16)
 (store ?s sm16) (name fred16 fred))
```

17. "John packed a suitcase.
    He paid the taxi-driver."

```
((inst pack17 packing) (agent-pack pack17 john17) (name john17 john)
 (patient-pack pack17 sc17) (inst sc17 suitcase) (precede pack17 pay17)
 (inst pay17 paying) (payer pay17 john17) (payee pay17 td17)
 (occupation td17 taxidriver))

((inst ?p going-by-plane) (inst ?t going-by-taxi) (pack-step ?p pack17)
 (go-step ?p ?t) (pay-step ?t pay17) (goer ?p john17)
 (plane-luggage ?p sc17) (taxi-driver ?t td17) (inst sc17 suitcase)
 (name john17 john) (precede pack17 pay17))
```

18. "Mary paid the taxi-driver.
    She ordered a milkshake."

```
((inst pay18 paying) (payer pay18 mary18) (name mary18 mary)
 (payee pay18 td18) (occupation td18 taxidriver) (precede pay18 order18)
 (inst order18 ordering) (agent-order order18 mary18)
 (patient-order order18 ms18) (inst ms18 milkshake))

((inst ?d rest-dining) (inst ?t going-by-taxi) (go-step ?d ?t)
 (pay-step ?t pay18) (order-step ?d order18) (diner ?d mary18)
 (taxi-driver ?t td18) (rest-thing-ordered ?d ms18)
 (inst ms18 milkshake) (name mary18 mary) (precede pay18 order18))
```

19. "Mary got a gun.
    She went to the supermarket.
    She found the milk on the shelf.
    She got some money from the cashier."

```
((inst get19 getting) (agent-get get19 mary19) (name mary19 mary)
 (patient-get get19 gun19) (inst gun19 gun) (precede get19 go19)
 (inst go19 going) (goer go19 mary19) (dest-go go19 sm19)
 (inst sm19 smarket) (precede go19 find19) (inst find19 finding)
 (finder find19 mary19) (thing-found find19 milk19) (inst milk19 milk)
 (on milk19 shf19) (inst shf19 shelf) (precede find19 get19b)
 (inst get19b getting) (agent-get get19b mary19)
 (patient-get get19b money19) (inst money19 money)
 (from-get get19b csh19) (occupation csh19 cashier))

((inst ?r robbing) (inst ?s smarket-shopping) (get-weapon-step ?r get19)
 (go-step ?r go19) (go-step ?s go19) (find-step ?s find19)
 (get-valuable-step ?r get19b) (robber ?r mary19) (shopper ?s mary19)
 (weapon-rob ?r gun19) (place-rob ?r sm19) (store ?s sm19)
 (thing-shopped-for ?s milk19) (thing-robbed ?r money19)
 (victim-rob ?r csh19) (inst gun19 gun) (inst milk19 milk)
 (on milk19 shf19) (inst shf19 shelf) (inst money19 money)
 (occupation csh19 cashier) (name mary19 mary) (precede get19 go19)
 (precede go19 find19) (precede find19 get19b))

20. "Bill took a taxi to the liquor store.
     He paid for the bourbon."

((inst go20 going) (goer go20 bill20) (name bill20 bill)
 (vehicle go20 taxi20) (inst taxi20 taxi) (dest-go go20 ls20)
 (inst ls20 liquor-store) (precede go20 pay20) (inst pay20 paying)
 (payer pay20 bill20) (thing-paid pay20 bourbon20)
 (inst bourbon20 bourbon))

((inst ?s liqst-shopping) (inst go20 going-by-taxi) (go-step ?s go20)
 (buy-step ?s ?b) (pay-step ?b pay20) (shopper ?s bill20)
 (vehicle go20 taxi20) (store ?s ls20) (thing-shopped-for ?s bourbon20)
 (inst bourbon20 bourbon)  (name bill20 bill) (precede go20 pay20))

21. "Bill paid for the bourbon."

((inst pay21 paying) (payer pay21 bill21) (name bill21 bill)
 (thing-paid pay21 bourbon21) (inst bourbon21 bourbon))

((inst pay21 paying) (payer pay21 bill21) (name bill21 bill)
 (thing-paid pay21 bourbon21) (inst bourbon21 bourbon))

22. "Bill paid for the milk."

((inst pay22 paying) (payer pay22 bill22) (name bill22 bill)
 (thing-paid pay22 milk22) (inst milk22 milk))

((inst pay22 paying) (payer pay22 bill22) (name bill22 bill)
 (thing-paid pay22 milk22) (inst milk22 milk))

23. "John bought a ticket.
     He went to the restaurant.
     He ordered a milkshake.
     He got on the plane."

((inst buy23 buying) (buyer buy23 john23) (name john23 john)
 (thing-bought buy23 tk23) (inst tk23 ticket) (precede buy23 go23)
 (inst go23 going) (goer go23 john23) (dest-go go23 rest23)
 (inst rest23 restaurant) (precede go23 order23) (inst order23 ordering)
 (agent-order order23 john23) (patient-order order23 ms23)
 (inst ms23 milkshake) (precede order23 geton23) (inst geton23 getting-on)
 (agent-get-on geton23 john23) (patient-get-on geton23 plane23)
 (inst plane23 plane))

((inst ?p going-by-plane) (buy-ticket-step ?p buy23)
 (get-on-step ?p geton23) (goer ?p john23) (plane-ticket ?p tk23)
 (vehicle ?p plane23) (inst ?d rest-dining) (go-step ?d go23)
 (order-step ?d order23) (diner ?d john23) (restaurant ?d rest23)
 (rest-thing-ordered ?d ms23) (inst ms23 milkshake) (name john23 john)
 (precede buy23 go23) (precede go23 order23) (precede order23 geton23))
```

```
24. "Mary went to the bus-station.
     She got off at the restaurant.
     She drank some milk."
```

```
((inst go24 going) (goer go24 mary24) (name mary24 mary)
 (dest-go go24 bus-station24) (inst bus-station24 bus-station)
 (precede go24 getoff24) (inst getoff24 getting-off)
 (agent-get-off getoff24 mary24) (place-get-off getoff24 rest24)
 (inst rest24 restaurant) (precede getoff24 drink24) (inst drink24 drinking)
 (drinker drink24 mary24) (patient-drink drink24 milk24) (inst milk24 milk))

((inst ?d rest-dining) (inst ?b going-by-bus) (go-step ?d ?b)
 (go-step ?b go24) (get-off-step ?b getoff24) (drink-step ?d drink24)
 (diner ?d mary24) (source-go ?b bus-station24) (restaurant ?d rest24)
 (rest-thing-drunk ?d milk24) (inst milk24 milk) (name mary24 mary)
 (precede go24 getoff24) (precede getoff24 drink24))
```

```
25. "Jane packed her bag.
     She got off the plane."
```

```
((inst pack25 packing) (agent-pack pack25 jane25) (name jane25 jane)
 (patient-pack pack25 bag25) (inst bag25 bag) (precede pack25 getoff25)
 (inst getoff25 getting-off) (agent-get-off getoff25 jane25)
 (patient-get-off getoff25 plane25) (inst plane25 plane))

((inst ?p going-by-plane) (pack-step ?p pack25) (get-off-step ?p getoff25)
 (goer ?p jane25) (plane-luggage ?p bag25) (vehicle ?p plane25)
 (name jane25 jane) (precede pack25 getoff25))
```

Appendix B

Set Covering Diagnosis

B.1 Knowledge Base

```
(setf *brules* '(
; right lateral medulla

((nystagmus-type gazeev) <- (right-lateral-medulla present))
((nystagmus-type horiz-left) <- (right-lateral-medulla present))
((nystagmus-type horiz-right) <- (right-lateral-medulla present))
((nystagmus-type vertical-upbeat) <- (right-lateral-medulla present))
((nystagmus-type vertical-downbeat) <- (right-lateral-medulla present))
((nystagmus-type rotatory) <- (right-lateral-medulla present))
((abnpupils-side-type right-miosis) <- (right-lateral-medulla present))
((ptosis-side right) <- (right-lateral-medulla present))
((facenumb-side right) <- (right-lateral-medulla present))
((facenumb-side left) <- (right-lateral-medulla present))
((swallow-severity partial) <- (right-lateral-medulla present))
((swallow-severity unable) <- (right-lateral-medulla present))
((gag-severity impaired) <- (right-lateral-medulla present))
((gag-severity absent) <- (right-lateral-medulla present))
((ataxia-type truncal) <- (right-lateral-medulla present))
((ataxia-type limb-right-mild) <- (right-lateral-medulla present))
((ataxia-type limb-right-severe) <- (right-lateral-medulla present))
((decram-side right-mild) <- (right-lateral-medulla present))
((decram-side right-severe) <- (right-lateral-medulla present))
((gait-type unsteady) <- (right-lateral-medulla present))
((gait-type other) <- (right-lateral-medulla present))
((pp-side left-mild) <- (right-lateral-medulla present))
((pp-side left-moderate) <- (right-lateral-medulla present))
((pp-side left-severe) <- (right-lateral-medulla present))
((temp-side left) <- (right-lateral-medulla present))

; left lateral medulla

((nystagmus-type gazeev) <- (left-lateral-medulla present))
((nystagmus-type horiz-left) <- (left-lateral-medulla present))
((nystagmus-type horiz-right) <- (left-lateral-medulla present))
((nystagmus-type vertical-upbeat) <- (left-lateral-medulla present))
((nystagmus-type vertical-downbeat) <- (left-lateral-medulla present))
((nystagmus-type rotatory) <- (left-lateral-medulla present))
((abnpupils-side-type left-miosis) <- (left-lateral-medulla present))
((ptosis-side left) <- (left-lateral-medulla present))
((facenumb-side right) <- (left-lateral-medulla present))
```

```
((facenumb-side left) <- (left-lateral-medulla present))
((swallow-severity partial) <- (left-lateral-medulla present))
((swallow-severity unable) <- (left-lateral-medulla present))
((gag-severity impaired) <- (left-lateral-medulla present))
((gag-severity absent) <- (left-lateral-medulla present))
((ataxia-type truncal) <- (left-lateral-medulla present))
((ataxia-type limb-left-mild) <- (left-lateral-medulla present))
((ataxia-type limb-left-severe) <- (left-lateral-medulla present))
((decram-side left-mild) <- (left-lateral-medulla present))
((decram-side left-severe) <- (left-lateral-medulla present))
((gait-type unsteady) <- (left-lateral-medulla present))
((gait-type other) <- (left-lateral-medulla present))
((pp-side right-mild) <- (left-lateral-medulla present))
((pp-side right-moderate) <- (left-lateral-medulla present))
((pp-side right-severe) <- (left-lateral-medulla present))
((temp-side right) <- (left-lateral-medulla present))

; right medial medulla

((nystagmus-type gazeev) <- (right-medial-medulla present))
((nystagmus-type vertical-upbeat) <- (right-medial-medulla present))
((nystagmus-type vertical-downbeat) <- (right-medial-medulla present))
((tongweak-side right) <- (right-medial-medulla present))
((tongweak-side left) <- (right-medial-medulla present))
((dysarthria-severity mild) <- (right-medial-medulla present))
((dysarthria-severity moderate) <- (right-medial-medulla present))
((dysarthria-severity severe) <- (right-medial-medulla present))
((weakness-type hemiparesis-left) <- (right-medial-medulla present))
((abndtrs-side left-incdtr) <- (right-medial-medulla present))
((abndtrs-side left-decdtr) <- (right-medial-medulla present))
((babs-side left) <- (right-medial-medulla present))
((gait-type lhemi) <- (right-medial-medulla present))
((gait-type unsteady) <- (right-medial-medulla present))
((gait-type other) <- (right-medial-medulla present))
((touch-side left) <- (right-medial-medulla present))
((posloss-side left) <- (right-medial-medulla present))
((vibloss-side left) <- (right-medial-medulla present))
((twopoint-side left) <- (right-medial-medulla present))
((agraph-side left) <- (right-medial-medulla present))

; left medial medulla

((nystagmus-type gazeev) <- (left-medial-medulla present))
((nystagmus-type vertical-upbeat) <- (left-medial-medulla present))
((nystagmus-type vertical-downbeat) <- (left-medial-medulla present))
((tongweak-side right) <- (left-medial-medulla present))
((tongweak-side left) <- (left-medial-medulla present))
((dysarthria-severity mild) <- (left-medial-medulla present))
((dysarthria-severity moderate) <- (left-medial-medulla present))
((dysarthria-severity severe) <- (left-medial-medulla present))
((weakness-type hemiparesis-right) <- (left-medial-medulla present))
((abndtrs-side right-incdtr) <- (left-medial-medulla present))
((abndtrs-side right-decdtr) <- (left-medial-medulla present))
((babs-side right) <- (left-medial-medulla present))
```

```
((gait-type rhemi) <- (left-medial-medulla present))
((gait-type unsteady) <- (left-medial-medulla present))
((gait-type other) <- (left-medial-medulla present))
((touch-side right) <- (left-medial-medulla present))
((posloss-side right) <- (left-medial-medulla present))
((vibloss-side right) <- (left-medial-medulla present))
((twopoint-side right) <- (left-medial-medulla present))
((agraph-side right) <- (left-medial-medulla present))

; right pons

((decloc-degree drowsy) <- (right-pons present))
((decloc-degree stupor) <- (right-pons present))
((decloc-degree coma) <- (right-pons present))
((abnpupils-side-type right-miosis) <- (right-pons present))
((abnpupils-side-type left-miosis) <- (right-pons present))
((ptosis-side right) <- (right-pons present))
((nystagmus-type gazeev) <- (right-pons present))
((nystagmus-type horiz-left) <- (right-pons present))
((nystagmus-type horiz-right) <- (right-pons present))
((nystagmus-type vertical-upbeat) <- (right-pons present))
((nystagmus-type rotatory) <- (right-pons present))
((abneom-type hgaze-right) <- (right-pons present))
((abneom-type ino-right) <- (right-pons present))
((abneom-type sixthn-right) <- (right-pons present))
((poorokn-direction lhoriz) <- (right-pons present))
((poorokn-direction rhoriz) <- (right-pons present))
((facenumb-side right) <- (right-pons present))
((facial-side-type right-peripheral) <- (right-pons present))
((facial-side-type left-central) <- (right-pons present))
((swallow-severity partial) <- (right-pons present))
((swallow-severity unable) <- (right-pons present))
((gag-severity impaired) <- (right-pons present))
((gag-severity absent) <- (right-pons present))
((dysarthria-severity mild) <- (right-pons present))
((dysarthria-severity moderate) <- (right-pons present))
((dysarthria-severity severe) <- (right-pons present))
((tongweak-side left) <- (right-pons present))
((ataxia-type limb-right-mild) <- (right-pons present))
((ataxia-type limb-right-severe) <- (right-pons present))
((weakness-type hemiparesis-left) <- (right-pons present))
((babs-side left) <- (right-pons present))
((abndtrs-side left-incdtr) <- (right-pons present))
((abndtrs-side left-decdtr) <- (right-pons present))
((gait-type lhemi) <- (right-pons present))
((gait-type unsteady) <- (right-pons present))
((gait-type other) <- (right-pons present))
((pp-side left-mild) <- (right-pons present))
((pp-side left-moderate) <- (right-pons present))
((pp-side left-severe) <- (right-pons present))
((touch-side left) <- (right-pons present))
((temp-side left) <- (right-pons present))
((posloss-side left) <- (right-pons present))
((vibloss-side left) <- (right-pons present))
```

```
((twopoint-side left) <- (right-pons present))
((agraph-side left) <- (right-pons present))

; left pons

((decloc-degree drowsy) <- (left-pons present))
((decloc-degree stupor) <- (left-pons present))
((decloc-degree coma) <- (left-pons present))
((abnpupils-side-type right-miosis) <- (left-pons present))
((abnpupils-side-type left-miosis) <- (left-pons present))
((ptosis-side left) <- (left-pons present))
((nystagmus-type gazeev) <- (left-pons present))
((nystagmus-type horiz-left) <- (left-pons present))
((nystagmus-type horiz-right) <- (left-pons present))
((nystagmus-type vertical-upbeat) <- (left-pons present))
((nystagmus-type rotatory) <- (left-pons present))
((abneom-type hgaze-left) <- (left-pons present))
((abneom-type ino-left) <- (left-pons present))
((abneom-type sixthn-left) <- (left-pons present))
((poorokn-direction lhoriz) <- (left-pons present))
((poorokn-direction rhoriz) <- (left-pons present))
((facenumb-side left) <- (left-pons present))
((facial-side-type left-peripheral) <- (left-pons present))
((facial-side-type right-central) <- (left-pons present))
((swallow-severity partial) <- (left-pons present))
((swallow-severity unable) <- (left-pons present))
((gag-severity impaired) <- (left-pons present))
((gag-severity absent) <- (left-pons present))
((dysarthria-severity mild) <- (left-pons present))
((dysarthria-severity moderate) <- (left-pons present))
((dysarthria-severity severe) <- (left-pons present))
((tongweak-side right) <- (left-pons present))
((ataxia-type limb-left-mild) <- (left-pons present))
((ataxia-type limb-left-severe) <- (left-pons present))
((weakness-type hemiparesis-right) <- (left-pons present))
((babs-side right) <- (left-pons present))
((abndtrs-side right-incdtr) <- (left-pons present))
((abndtrs-side right-decdtr) <- (left-pons present))
((gait-type rhemi) <- (left-pons present))
((gait-type unsteady) <- (left-pons present))
((gait-type other) <- (left-pons present))
((pp-side right-mild) <- (left-pons present))
((pp-side right-moderate) <- (left-pons present))
((pp-side right-severe) <- (left-pons present))
((touch-side right) <- (left-pons present))
((temp-side right) <- (left-pons present))
((posloss-side right) <- (left-pons present))
((vibloss-side right) <- (left-pons present))
((twopoint-side right) <- (left-pons present))
((agraph-side right) <- (left-pons present))

; right midbrain

((decloc-degree coma) <- (right-midbrain present))
```

```
((decloc-degree drowsy) <- (right-midbrain present))
((decloc-degree stupor) <- (right-midbrain present))
((disoriented-degree mild) <- (right-midbrain present))
((disoriented-degree moderate) <- (right-midbrain present))
((disoriented-degree severe) <- (right-midbrain present))
((cogabn present) <- (right-midbrain present))
((nystagmus-type gazeev) <- (right-midbrain present))
((nystagmus-type horiz-left) <- (right-midbrain present))
((nystagmus-type horiz-right) <- (right-midbrain present))
((nystagmus-type vertical-upbeat) <- (right-midbrain present))
((nystagmus-type vertical-downbeat) <- (right-midbrain present))
((nystagmus-type rotatory) <- (right-midbrain present))
((poorokn-direction lhoriz) <- (right-midbrain present))
((poorokn-direction vertical) <- (right-midbrain present))
((abnpupils-side-type right-mydriasis) <- (right-midbrain present))
((abnpupils-side-type right-miosis) <- (right-midbrain present))
((prd-side right) <- (right-midbrain present))
((abneom-type vgaze-up) <- (right-midbrain present))
((abneom-type vgaze-down) <- (right-midbrain present))
((abneom-type ino-right) <- (right-midbrain present))
((abneom-type thirdn-right) <- (right-midbrain present))
((abneom-type skew) <- (right-midbrain present))
((ptosis-side right) <- (right-midbrain present))
((ptosis-side left) <- (right-midbrain present))
((facenumb-side left) <- (right-midbrain present))
((facial-side-type left-central) <- (right-midbrain present))
((swallow-severity partial) <- (right-midbrain present))
((swallow-severity unable) <- (right-midbrain present))
((gag-severity impaired) <- (right-midbrain present))
((gag-severity absent) <- (right-midbrain present))
((weakness-type hemiparesis-left) <- (right-midbrain present))
((ataxia-type limb-left-mild) <- (right-midbrain present))
((ataxia-type limb-left-severe) <- (right-midbrain present))
((ataxia-type limb-right-mild) <- (right-midbrain present))
((ataxia-type limb-right-severe) <- (right-midbrain present))
((decram-side left-mild) <- (right-midbrain present))
((decram-side left-severe) <- (right-midbrain present))
((babs-side left) <- (right-midbrain present))
((abndtrs-side left-incdtr) <- (right-midbrain present))
((abndtrs-side left-decdtr) <- (right-midbrain present))
((gait-type lhemi) <- (right-midbrain present))
((gait-type unsteady) <- (right-midbrain present))
((gait-type other) <- (right-midbrain present))
((dss-side left) <- (right-midbrain present))
((pp-side left-mild) <- (right-midbrain present))
((pp-side left-moderate) <- (right-midbrain present))
((pp-side left-severe) <- (right-midbrain present))
((touch-side left) <- (right-midbrain present))
((temp-side left) <- (right-midbrain present))
((posloss-side left) <- (right-midbrain present))
((vibloss-side left) <- (right-midbrain present))
((twopoint-side left) <- (right-midbrain present))
((agraph-side left) <- (right-midbrain present))
```

```
; left midbrain

((decloc-degree coma) <- (left-midbrain present))
((decloc-degree drowsy) <- (left-midbrain present))
((decloc-degree stupor) <- (left-midbrain present))
((disoriented-degree mild) <- (left-midbrain present))
((disoriented-degree moderate) <- (left-midbrain present))
((disoriented-degree severe) <- (left-midbrain present))
((cogabn present) <- (left-midbrain present))
((nystagmus-type gazeev) <- (left-midbrain present))
((nystagmus-type horiz-left) <- (left-midbrain present))
((nystagmus-type horiz-right) <- (left-midbrain present))
((nystagmus-type vertical-upbeat) <- (left-midbrain present))
((nystagmus-type vertical-downbeat) <- (left-midbrain present))
((nystagmus-type rotatory) <- (left-midbrain present))
((poorokn-direction rhoriz) <- (left-midbrain present))
((poorokn-direction vertical) <- (left-midbrain present))
((abnpupils-side-type left-mydriasis) <- (left-midbrain present))
((abnpupils-side-type left-miosis) <- (left-midbrain present))
((prd-side left) <- (left-midbrain present))
((abneom-type vgaze-up) <- (left-midbrain present))
((abneom-type vgaze-down) <- (left-midbrain present))
((abneom-type ino-left) <- (left-midbrain present))
((abneom-type thirdn-left) <- (left-midbrain present))
((abneom-type skew) <- (left-midbrain present))
((ptosis-side right) <- (left-midbrain present))
((ptosis-side left) <- (left-midbrain present))
((facenumb-side right) <- (left-midbrain present))
((facial-side-type right-central) <- (left-midbrain present))
((swallow-severity partial) <- (left-midbrain present))
((swallow-severity unable) <- (left-midbrain present))
((gag-severity impaired) <- (left-midbrain present))
((gag-severity absent) <- (left-midbrain present))
((weakness-type hemiparesis-right) <- (left-midbrain present))
((ataxia-type limb-left-mild) <- (left-midbrain present))
((ataxia-type limb-left-severe) <- (left-midbrain present))
((ataxia-type limb-right-mild) <- (left-midbrain present))
((ataxia-type limb-right-severe) <- (left-midbrain present))
((decram-side right-mild) <- (left-midbrain present))
((decram-side right-severe) <- (left-midbrain present))
((babs-side right) <- (left-midbrain present))
((abndtrs-side right-incdtr) <- (left-midbrain present))
((abndtrs-side right-decdtr) <- (left-midbrain present))
((gait-type rhemi) <- (left-midbrain present))
((gait-type unsteady) <- (left-midbrain present))
((gait-type other) <- (left-midbrain present))
((dss-side right) <- (left-midbrain present))
((pp-side right-mild) <- (left-midbrain present))
((pp-side right-moderate) <- (left-midbrain present))
((pp-side right-severe) <- (left-midbrain present))
((touch-side right) <- (left-midbrain present))
((temp-side right) <- (left-midbrain present))
((posloss-side right) <- (left-midbrain present))
((vibloss-side right) <- (left-midbrain present))
```

```
((twopoint-side right) <- (left-midbrain present))
((agraph-side right) <- (left-midbrain present))

; right cerebellar hemisphere

((nystagmus-type gazeev) <- (right-cerebellar-hemisphere present))
((nystagmus-type horiz-left) <- (right-cerebellar-hemisphere present))
((nystagmus-type horiz-right) <- (right-cerebellar-hemisphere present))
((nystagmus-type rotatory) <- (right-cerebellar-hemisphere present))
((dysarthria-severity mild) <- (right-cerebellar-hemisphere present))
((dysarthria-severity moderate) <- (right-cerebellar-hemisphere present))
((dysarthria-severity severe) <- (right-cerebellar-hemisphere present))
((ataxia-type limb-right-mild) <- (right-cerebellar-hemisphere present))
((ataxia-type limb-right-severe) <- (right-cerebellar-hemisphere present))
((decram-side right-mild) <- (right-cerebellar-hemisphere present))
((decram-side right-severe) <- (right-cerebellar-hemisphere present))
((gait-type unsteady) <- (right-cerebellar-hemisphere present))
((gait-type other) <- (right-cerebellar-hemisphere present))

; left cerebellar hemisphere

((nystagmus-type gazeev) <- (left-cerebellar-hemisphere present))
((nystagmus-type horiz-left) <- (left-cerebellar-hemisphere present))
((nystagmus-type horiz-right) <- (left-cerebellar-hemisphere present))
((nystagmus-type rotatory) <- (left-cerebellar-hemisphere present))
((dysarthria-severity mild) <- (left-cerebellar-hemisphere present))
((dysarthria-severity moderate) <- (left-cerebellar-hemisphere present))
((dysarthria-severity severe) <- (left-cerebellar-hemisphere present))
((ataxia-type limb-left-mild) <- (left-cerebellar-hemisphere present))
((ataxia-type limb-left-severe) <- (left-cerebellar-hemisphere present))
((decram-side left-mild) <- (left-cerebellar-hemisphere present))
((decram-side left-severe) <- (left-cerebellar-hemisphere present))
((gait-type unsteady) <- (left-cerebellar-hemisphere present))
((gait-type other) <- (left-cerebellar-hemisphere present))

; cerebellar vermis

((nystagmus-type gazeev) <- (cerebellar-vermis present))
((nystagmus-type horiz-left) <- (cerebellar-vermis present))
((nystagmus-type horiz-right) <- (cerebellar-vermis present))
((nystagmus-type vertical-upbeat) <- (cerebellar-vermis present))
((nystagmus-type vertical-downbeat) <- (cerebellar-vermis present))
((nystagmus-type rotatory) <- (cerebellar-vermis present))
((ataxia-type truncal) <- (cerebellar-vermis present))
((gait-type unsteady) <- (cerebellar-vermis present))

; right thalamus

((vf-deficit-side-type left-hemianopsia) <- (right-thalamus present))
((vf-deficit-side-type left-quadrantanopsia-inferior)
  <- (right-thalamus present))
((vf-deficit-side-type left-quadrantanopsia-superior)
  <- (right-thalamus present))
((facenumb-side left) <- (right-thalamus present))
```

```
((abneom-type hgaze-left) <- (right-thalamus present))
((abneom-type hgaze-right) <- (right-thalamus present))
((ataxia-type limb-left-mild) <- (right-thalamus present))
((ataxia-type limb-left-severe) <- (right-thalamus present))
((pp-side left-mild) <- (right-thalamus present))
((pp-side left-moderate) <- (right-thalamus present))
((pp-side left-severe) <- (right-thalamus present))
((touch-side left) <- (right-thalamus present))
((temp-side left) <- (right-thalamus present))
((posloss-side left) <- (right-thalamus present))
((vibloss-side left) <- (right-thalamus present))
((twopoint-side left) <- (right-thalamus present))
((agraph-side left) <- (right-thalamus present))

; left thalamus

((anomia-severity mild) <- (left-thalamus present))
((anomia-severity moderate) <- (left-thalamus present))
((anomia-severity severe) <- (left-thalamus present))
((compdef-severity mild) <- (left-thalamus present))
((compdef-severity moderate) <- (left-thalamus present))
((compdef-severity severe) <- (left-thalamus present))
((vf-deficit-side-type left-hemianopsia) <- (left-thalamus present))
((vf-deficit-side-type left-quadrantanopsia-inferior)
  <- (left-thalamus present))
((vf-deficit-side-type left-quadrantanopsia-superior)
  <- (left-thalamus present))
((facenumb-side right) <- (left-thalamus present))
((abneom-type hgaze-left) <- (left-thalamus present))
((abneom-type hgaze-right) <- (left-thalamus present))
((ataxia-type limb-right-mild) <- (left-thalamus present))
((ataxia-type limb-right-severe) <- (left-thalamus present))
((pp-side right-mild) <- (left-thalamus present))
((pp-side right-moderate) <- (left-thalamus present))
((pp-side right-severe) <- (left-thalamus present))
((touch-side right) <- (left-thalamus present))
((temp-side right) <- (left-thalamus present))
((posloss-side right) <- (left-thalamus present))
((vibloss-side right) <- (left-thalamus present))
((twopoint-side right) <- (left-thalamus present))
((agraph-side right) <- (left-thalamus present))

; right basal ganglia

((decloc-degree drowsy) <- (right-basal-ganglia present))
((cogabn present) <- (right-basal-ganglia present))
((decram-side left-mild) <- (right-basal-ganglia present))
((decram-side left-severe) <- (right-basal-ganglia present))
((gait-type other) <- (right-basal-ganglia present))

; left basal ganglia

((decloc-degree drowsy) <- (left-basal-ganglia present))
((cogabn present) <- (left-basal-ganglia present))
```

```
((decram-side right-mild) <- (left-basal-ganglia present))
((decram-side right-severe) <- (left-basal-ganglia present))
((gait-type other) <- (left-basal-ganglia present))

; right internal capsule

((facenumb-side left) <- (right-internal-capsule present))
((facial-side-type left-central) <- (right-internal-capsule present))
((tongweak-side left) <- (right-internal-capsule present))
((swallow-severity partial) <- (right-internal-capsule present))
((swallow-severity unable) <- (right-internal-capsule present))
((gag-severity impaired) <- (right-internal-capsule present))
((gag-severity absent) <- (right-internal-capsule present))
((dysarthria-severity mild) <- (right-internal-capsule present))
((dysarthria-severity moderate) <- (right-internal-capsule present))
((dysarthria-severity severe) <- (right-internal-capsule present))
((weakness-type hemiparesis-left) <- (right-internal-capsule present))
((weakness-type monoparesis-lue) <- (right-internal-capsule present))
((weakness-type monoparesis-lle) <- (right-internal-capsule present))
((decram-side left-mild) <- (right-internal-capsule present))
((decram-side left-severe) <- (right-internal-capsule present))
((babs-side left) <- (right-internal-capsule present))
((abndtrs-side left-incdtr) <- (right-internal-capsule present))
((abndtrs-side left-decdtr) <- (right-internal-capsule present))
((gait-type lhemi) <- (right-internal-capsule present))
((gait-type other) <- (right-internal-capsule present))
((dss-side left) <- (right-internal-capsule present))
((pp-side left-mild) <- (right-internal-capsule present))
((pp-side left-moderate) <- (right-internal-capsule present))
((pp-side left-severe) <- (right-internal-capsule present))
((touch-side left) <- (right-internal-capsule present))
((temp-side left) <- (right-internal-capsule present))
((posloss-side left) <- (right-internal-capsule present))
((vibloss-side left) <- (right-internal-capsule present))
((twopoint-side left) <- (right-internal-capsule present))
((agraph-side left) <- (right-internal-capsule present))

; left internal capsule

((facenumb-side right) <- (left-internal-capsule present))
((facial-side-type right-central) <- (left-internal-capsule present))
((tongweak-side right) <- (left-internal-capsule present))
((swallow-severity partial) <- (left-internal-capsule present))
((swallow-severity unable) <- (left-internal-capsule present))
((gag-severity impaired) <- (left-internal-capsule present))
((gag-severity absent) <- (left-internal-capsule present))
((dysarthria-severity mild) <- (left-internal-capsule present))
((dysarthria-severity moderate) <- (left-internal-capsule present))
((dysarthria-severity severe) <- (left-internal-capsule present))
((weakness-type hemiparesis-right) <- (left-internal-capsule present))
((weakness-type monoparesis-rue) <- (left-internal-capsule present))
((weakness-type monoparesis-rle) <- (left-internal-capsule present))
((decram-side right-mild) <- (left-internal-capsule present))
((decram-side right-severe) <- (left-internal-capsule present))
```

```
((babs-side right) <- (left-internal-capsule present))
((abndtrs-side right-incdtr) <- (left-internal-capsule present))
((abndtrs-side right-decdtr) <- (left-internal-capsule present))
((gait-type rhemi) <- (left-internal-capsule present))
((gait-type other) <- (left-internal-capsule present))
((dss-side right) <- (left-internal-capsule present))
((pp-side right-mild) <- (left-internal-capsule present))
((pp-side right-moderate) <- (left-internal-capsule present))
((pp-side right-severe) <- (left-internal-capsule present))
((touch-side right) <- (left-internal-capsule present))
((temp-side right) <- (left-internal-capsule present))
((posloss-side right) <- (left-internal-capsule present))
((vibloss-side right) <- (left-internal-capsule present))
((twopoint-side right) <- (left-internal-capsule present))
((agraph-side right) <- (left-internal-capsule present))

; right frontal lobe

((decloc-degree drowsy) <- (right-frontal-lobe present))
((decloc-degree stupor) <- (right-frontal-lobe present))
((decloc-degree coma) <- (right-frontal-lobe present))
((disoriented-degree mild) <- (right-frontal-lobe present))
((disoriented-degree moderate) <- (right-frontal-lobe present))
((disoriented-degree severe) <- (right-frontal-lobe present))
((dyspraxia present) <- (right-frontal-lobe present))
((hemineglect-side left) <- (right-frontal-lobe present))
((denial present) <- (right-frontal-lobe present))
((compdef-severity mild) <- (right-frontal-lobe present))
((nonfluency-severity mild) <- (right-frontal-lobe present))
((nonfluency-severity moderate) <- (right-frontal-lobe present))
((nonfluency-severity severe) <- (right-frontal-lobe present))
((repetition-severity mild) <- (right-frontal-lobe present))
((repetition-severity moderate) <- (right-frontal-lobe present))
((repetition-severity severe) <- (right-frontal-lobe present))
((anomia-severity mild) <- (right-frontal-lobe present))
((anomia-severity moderate) <- (right-frontal-lobe present))
((anomia-severity severe) <- (right-frontal-lobe present))
((cogabn present) <- (right-frontal-lobe present))
((dysarthria-severity mild) <- (right-frontal-lobe present))
((dysarthria-severity moderate) <- (right-frontal-lobe present))
((dysarthria-severity severe) <- (right-frontal-lobe present))
((poorokn-direction lhoriz) <- (right-frontal-lobe present))
((nystagmus-type gazeev) <- (right-frontal-lobe present))
((abneom-type hgaze-left) <- (right-frontal-lobe present))
((facial-side-type left-central) <- (right-frontal-lobe present))
((swallow-severity partial) <- (right-frontal-lobe present))
((swallow-severity unable) <- (right-frontal-lobe present))
((gag-severity impaired) <- (right-frontal-lobe present))
((gag-severity absent) <- (right-frontal-lobe present))
((tongweak-side left) <- (right-frontal-lobe present))
((weakness-type hemiparesis-left) <- (right-frontal-lobe present))
((weakness-type monoparesis-lue) <- (right-frontal-lobe present))
((weakness-type monoparesis-lle) <- (right-frontal-lobe present))
((decram-side left-mild) <- (right-frontal-lobe present))
```

```
((decram-side left-severe) <- (right-frontal-lobe present))
((gait-type lhemi) <- (right-frontal-lobe present))
((gait-type unsteady) <- (right-frontal-lobe present))
((gait-type other) <- (right-frontal-lobe present))
((ataxia-type limb-left-mild) <- (right-frontal-lobe present))
((abndtrs-side left-incdtr) <- (right-frontal-lobe present))
((abndtrs-side left-decdtr) <- (right-frontal-lobe present))
((babs-side left) <- (right-frontal-lobe present))

; left frontal lobe

((decloc-degree drowsy) <- (left-frontal-lobe present))
((decloc-degree stupor) <- (left-frontal-lobe present))
((decloc-degree coma) <- (left-frontal-lobe present))
((disoriented-degree mild) <- (left-frontal-lobe present))
((disoriented-degree moderate) <- (left-frontal-lobe present))
((disoriented-degree severe) <- (left-frontal-lobe present))
((dyspraxia present) <- (left-frontal-lobe present))
((hemineglect-side right) <- (left-frontal-lobe present))
((denial present) <- (left-frontal-lobe present))
((compdef-severity mild) <- (left-frontal-lobe present))
((compdef-severity moderate) <- (left-frontal-lobe present))
((compdef-severity severe) <- (left-frontal-lobe present))
((nonfluency-severity mild) <- (left-frontal-lobe present))
((nonfluency-severity moderate) <- (left-frontal-lobe present))
((nonfluency-severity severe) <- (left-frontal-lobe present))
((repetition-severity mild) <- (left-frontal-lobe present))
((repetition-severity moderate) <- (left-frontal-lobe present))
((repetition-severity severe) <- (left-frontal-lobe present))
((anomia-severity mild) <- (left-frontal-lobe present))
((anomia-severity moderate) <- (left-frontal-lobe present))
((anomia-severity severe) <- (left-frontal-lobe present))
((cogabn present) <- (left-frontal-lobe present))
((dysarthria-severity mild) <- (left-frontal-lobe present))
((dysarthria-severity moderate) <- (left-frontal-lobe present))
((dysarthria-severity severe) <- (left-frontal-lobe present))
((poorokn-direction rhoriz) <- (left-frontal-lobe present))
((nystagmus-type horiz-right) <- (left-frontal-lobe present))
((abneom-type hgaze-right) <- (left-frontal-lobe present))
((facial-side-type right-central) <- (left-frontal-lobe present))
((swallow-severity partial) <- (left-frontal-lobe present))
((swallow-severity unable) <- (left-frontal-lobe present))
((gag-severity impaired) <- (left-frontal-lobe present))
((gag-severity absent) <- (left-frontal-lobe present))
((tongweak-side right) <- (left-frontal-lobe present))
((weakness-type hemiparesis-right) <- (left-frontal-lobe present))
((weakness-type monoparesis-lue) <- (left-frontal-lobe present))
((weakness-type monoparesis-lle) <- (left-frontal-lobe present))
((decram-side right-mild) <- (left-frontal-lobe present))
((decram-side right-severe) <- (left-frontal-lobe present))
((gait-type rhemi) <- (left-frontal-lobe present))
((gait-type unsteady) <- (left-frontal-lobe present))
((gait-type other) <- (left-frontal-lobe present))
((ataxia-type limb-right-mild) <- (left-frontal-lobe present))
```

```
((abndtrs-side right-incdtr) <- (left-frontal-lobe present))
((abndtrs-side right-decdtr) <- (left-frontal-lobe present))
((babs-side right) <- (left-frontal-lobe present))

; right parietal lobe

((decloc-degree drowsy) <- (right-parietal-lobe present))
((decloc-degree stupor) <- (right-parietal-lobe present))
((decloc-degree coma) <- (right-parietal-lobe present))
((disoriented-degree mild) <- (right-parietal-lobe present))
((disoriented-degree moderate) <- (right-parietal-lobe present))
((disoriented-degree severe) <- (right-parietal-lobe present))
((dyspraxia present) <- (right-parietal-lobe present))
((denial present) <- (right-parietal-lobe present))
((hemineglect-side left) <- (right-parietal-lobe present))
((cogabn present) <- (right-parietal-lobe present))
((vf-deficit-side-type left-hemianopsia) <- (right-parietal-lobe present))
((vf-deficit-side-type left-quadrantanopsia-inferior)
  <- (right-parietal-lobe present))
((poorokn-direction lhoriz) <- (right-parietal-lobe present))
((nystagmus-type gazeev) <- (right-parietal-lobe present))
((abneom-type hgaze-left) <- (right-parietal-lobe present))
((facenumb-side left) <- (right-parietal-lobe present))
((swallow-severity partial) <- (right-parietal-lobe present))
((swallow-severity unable) <- (right-parietal-lobe present))
((gag-severity impaired) <- (right-parietal-lobe present))
((gag-severity absent) <- (right-parietal-lobe present))
((gait-type lhemi) <- (right-parietal-lobe present))
((gait-type other) <- (right-parietal-lobe present))
((dss-side left) <- (right-parietal-lobe present))
((pp-side left-mild) <- (right-parietal-lobe present))
((pp-side left-moderate) <- (right-parietal-lobe present))
((pp-side left-severe) <- (right-parietal-lobe present))
((posloss-side left) <- (right-parietal-lobe present))
((vibloss-side left) <- (right-parietal-lobe present))
((twopoint-side left) <- (right-parietal-lobe present))
((agraph-side left) <- (right-parietal-lobe present))

; left parietal lobe

((decloc-degree drowsy) <- (left-parietal-lobe present))
((decloc-degree stupor) <- (left-parietal-lobe present))
((decloc-degree coma) <- (left-parietal-lobe present))
((disoriented-degree mild) <- (left-parietal-lobe present))
((disoriented-degree moderate) <- (left-parietal-lobe present))
((disoriented-degree severe) <- (left-parietal-lobe present))
((hemineglect-side right) <- (left-parietal-lobe present))
((compdef-severity mild) <- (left-parietal-lobe present))
((compdef-severity moderate) <- (left-parietal-lobe present))
((compdef-severity severe) <- (left-parietal-lobe present))
((repetition-severity mild) <- (left-parietal-lobe present))
((repetition-severity moderate) <- (left-parietal-lobe present))
((repetition-severity severe) <- (left-parietal-lobe present))
((nonfluency-severity mild) <- (left-parietal-lobe present))
```

```
((nonfluency-severity moderate) <- (left-parietal-lobe present))
((nonfluency-severity severe) <- (left-parietal-lobe present))
((anomia-severity mild) <- (left-parietal-lobe present))
((anomia-severity moderate) <- (left-parietal-lobe present))
((anomia-severity severe) <- (left-parietal-lobe present))
((cogabn present) <- (left-parietal-lobe present))
((vf-deficit-side-type right-hemianopsia) <- (left-parietal-lobe present))
((vf-deficit-side-type right-quadrantanopsia-inferior)
  <- (left-parietal-lobe present))
((dysarthria-severity mild) <- (left-parietal-lobe present))
((dysarthria-severity moderate) <- (left-parietal-lobe present))
((dysarthria-severity severe) <- (left-parietal-lobe present))
((poorokn-direction rhoriz) <- (left-parietal-lobe present))
((nystagmus-type gazeev) <- (left-parietal-lobe present))
((abneom-type hgaze-right) <- (left-parietal-lobe present))
((facenumb-side right) <- (left-parietal-lobe present))
((swallow-severity partial) <- (left-parietal-lobe present))
((swallow-severity unable) <- (left-parietal-lobe present))
((gag-severity impaired) <- (left-parietal-lobe present))
((gag-severity absent) <- (left-parietal-lobe present))
((gait-type rhemi) <- (left-parietal-lobe present))
((gait-type other) <- (left-parietal-lobe present))
((dss-side right) <- (left-parietal-lobe present))
((pp-side right-mild) <- (left-parietal-lobe present))
((pp-side right-moderate) <- (left-parietal-lobe present))
((pp-side right-severe) <- (left-parietal-lobe present))
((posloss-side right) <- (left-parietal-lobe present))
((vibloss-side right) <- (left-parietal-lobe present))
((twopoint-side right) <- (left-parietal-lobe present))
((agraph-side right) <- (left-parietal-lobe present))

; right temporal lobe

((disoriented-degree mild) <- (right-temporal-lobe present))
((disoriented-degree moderate) <- (right-temporal-lobe present))
((disoriented-degree severe) <- (right-temporal-lobe present))
((decloc-degree drowsy) <- (right-temporal-lobe present))
((decloc-degree stupor) <- (right-temporal-lobe present))
((decloc-degree coma) <- (right-temporal-lobe present))
((cogabn present) <- (right-temporal-lobe present))
((hemineglect-side left) <- (right-temporal-lobe present))
((denial present) <- (right-temporal-lobe present))
((abneom-type hgaze-left) <- (right-temporal-lobe present))
((poorokn-direction lhoriz) <- (right-temporal-lobe present))
((swallow-severity partial) <- (right-temporal-lobe present))
((swallow-severity unable) <- (right-temporal-lobe present))
((gag-severity impaired) <- (right-temporal-lobe present))
((gag-severity absent) <- (right-temporal-lobe present))
((vf-deficit-side-type left-hemianopsia) <- (right-temporal-lobe present))
((vf-deficit-side-type left-quadrantanopsia-superior)
  <- (right-temporal-lobe present))

; left temporal lobe
```

```
((disoriented-degree mild) <- (left-temporal-lobe present))
((disoriented-degree moderate) <- (left-temporal-lobe present))
((disoriented-degree severe) <- (left-temporal-lobe present))
((decloc-degree drowsy) <- (left-temporal-lobe present))
((decloc-degree stupor) <- (left-temporal-lobe present))
((decloc-degree coma) <- (left-temporal-lobe present))
((hemineglect-side right) <- (left-temporal-lobe present))
((compdef-severity mild) <- (left-temporal-lobe present))
((compdef-severity moderate) <- (left-temporal-lobe present))
((compdef-severity severe) <- (left-temporal-lobe present))
((repetition-severity mild) <- (left-temporal-lobe present))
((repetition-severity moderate) <- (left-temporal-lobe present))
((repetition-severity severe) <- (left-temporal-lobe present))
((anomia-severity mild) <- (left-temporal-lobe present))
((anomia-severity moderate) <- (left-temporal-lobe present))
((anomia-severity severe) <- (left-temporal-lobe present))
((nonfluency-severity mild) <- (left-temporal-lobe present))
((nonfluency-severity moderate) <- (left-temporal-lobe present))
((nonfluency-severity severe) <- (left-temporal-lobe present))
((cogabn present) <- (left-temporal-lobe present))
((abneom-type hgaze-right) <- (left-temporal-lobe present))
((poorokn-direction rhoriz) <- (left-temporal-lobe present))
((swallow-severity partial) <- (left-temporal-lobe present))
((swallow-severity unable) <- (left-temporal-lobe present))
((gag-severity impaired) <- (left-temporal-lobe present))
((gag-severity absent) <- (left-temporal-lobe present))
((vf-deficit-side-type right-hemianopsia) <- (left-temporal-lobe present))
((vf-deficit-side-type right-quadrantanopsia-superior)
  <- (left-temporal-lobe present))

; right occipital lobe

((poorokn-direction lhoriz) <- (right-occipital-lobe present))
((vf-deficit-side-type left-hemianopsia) <- (right-occipital-lobe present))
((vf-deficit-side-type left-quadrantanopsia-superior)
  <- (right-occipital-lobe present))
((vf-deficit-side-type left-quadrantanopsia-inferior)
  <- (right-occipital-lobe present))

; left occipital lobe

((poorokn-direction rhoriz) <- (left-occipital-lobe present))
((vf-deficit-side-type right-hemianopsia) <- (left-occipital-lobe present))
((vf-deficit-side-type right-quadrantanopsia-superior)
  <- (left-occipital-lobe present))
((vf-deficit-side-type right-quadrantanopsia-inferior)
  <- (left-occipital-lobe present))
))

(setf *facts* nil)

(setf *nogoods* nil)
(setf *assumption-nogoods* nil)
```

```
(setf *inter-batch-beam-width* most-positive-fixnum)
(setf *intra-batch-beam-width* most-positive-fixnum)
(setf *bchain-depth* 2)
(setf *caching* t)
(setf *factoring* t)
(setf *remove-superset?* t)
(setf *remove-superset-fn* #'set-<=)
(setf *explanation-eval-metric* #'simplicity-only)

(setf *predicate-specific-abduction* t)
(setf *assumable-predicates*
      '(right-lateral-medulla left-lateral-medulla
        right-medial-medulla left-medial-medulla
        right-pons left-pons
        right-midbrain left-midbrain
        right-cerebellar-hemisphere left-cerebellar-hemisphere
        cerebellar-vermis
        right-thalamus left-thalamus
        right-basal-ganglia left-basal-ganglia
        right-internal-capsule left-internal-capsule
        right-frontal-lobe left-frontal-lobe
        right-parietal-lobe left-parietal-lobe
        right-temporal-lobe left-temporal-lobe
        right-occipital-lobe left-occipital-lobe))
(setf *free-assumption-predicates* nil)
```

B.2 Patient Cases

The 50 patient cases used to test ACCEL are listed below. For each case, we give (1) the observed symptoms (*$*s\#*$*); and (2) the diagnosis given by the physician (*$*d\#*$*).

```
*s1*   ((disoriented-degree mild) (compdef-severity mild)
        (nonfluency-severity mild) (facial-side-type right-central))
*d1*   ((left-frontal-lobe present))

*s2*   ((disoriented-degree moderate)
        (vf-deficit-side-type left-hemianopsia)
        (poorokn-direction lhoriz) (abneom-type hgaze-left)
        (facenumb-side left) (facial-side-type left-central)
        (dysarthria-severity moderate) (weakness-type hemiparesis-left)
        (abndtrs-side left-decdtr) (babs-side left)
        (pp-side left-mild))
*d2*   ((right-frontal-lobe present) (right-parietal-lobe present))

*s3*   ((dyspraxia present) (hemineglect-side right)
        (compdef-severity severe) (nonfluency-severity severe)
        (repetition-severity severe) (anomia-severity severe)
        (vf-deficit-side-type right-hemianopsia)
        (weakness-type hemiparesis-right) (decram-side right-mild))
*d3*   ((left-frontal-lobe present) (left-parietal-lobe present))
```

```
*s4*   ((abneom-type hgaze-right) (facial-side-type right-central)
        (dysarthria-severity moderate) (weakness-type hemiparesis-right)
        (abndtrs-side right-incdtr))
*d4*   ((left-frontal-lobe present) (left-internal-capsule present))

*s5*   ((decloc-degree drowsy) (hemineglect-side left) (denial present)
        (vf-deficit-side-type left-hemianopsia) (abneom-type hgaze-left)
        (facenumb-side left) (facial-side-type left-central)
        (dysarthria-severity mild) (weakness-type hemiparesis-left)
        (abndtrs-side left-incdtr) (dss-side left) (pp-side left-severe)
        (posloss-side left) (vibloss-side left) (twopoint-side left)
        (agraph-side left))
*d5*   ((right-frontal-lobe present) (right-parietal-lobe present)
        (right-temporal-lobe present))

*s6*   ((hemineglect-side left) (vf-deficit-side-type left-hemianopsia)
        (facial-side-type left-central) (weakness-type hemiparesis-left)
        (decram-side left-mild) (abndtrs-side left-decdtr)
        (dss-side left) (posloss-side left))
*d6*   ((right-frontal-lobe present) (right-parietal-lobe present))

*s7*   ((decloc-degree stupor) (abneom-type hgaze-left) (facenumb-side left)
        (facial-side-type left-central) (dysarthria-severity severe)
        (weakness-type hemiparesis-left) (abndtrs-side left-incdtr)
        (babs-side left) (pp-side left-moderate))
*d7*   ((right-frontal-lobe present) (right-parietal-lobe present)
        (right-temporal-lobe present))

*s8*   ((facenumb-side right) (weakness-type hemiparesis-right)
        (decram-side right-mild) (abndtrs-side right-incdtr)
        (babs-side right) (gait-type other) (pp-side right-mild)
        (vibloss-side right))
*d8*   ((left-internal-capsule present))

*s9*   ((vf-deficit-side-type left-hemianopsia) (dss-side left))
*d9*   ((right-parietal-lobe present) (right-occipital-lobe present))

*s10*  ((decloc-degree drowsy) (abneom-type hgaze-right)
        (facial-side-type right-central)
        (weakness-type hemiparesis-right) (decram-side right-mild)
        (abndtrs-side right-incdtr) (babs-side right))
*d10*  ((left-frontal-lobe present))

*s11*  ((decloc-degree drowsy) (disoriented-degree moderate)
        (hemineglect-side right) (abneom-type hgaze-right)
        (facenumb-side right) (dysarthria-severity moderate)
        (weakness-type hemiparesis-right) (abndtrs-side right-incdtr)
        (babs-side right) (pp-side right-mild))
*d11*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s12*  ((weakness-type hemiparesis-left) (abndtrs-side left-incdtr)
        (babs-side left))
*d12*  ((right-internal-capsule present))
```

```
*s13*  ((decloc-degree drowsy) (disoriented-degree severe)
        (hemineglect-side left) (denial present)
        (vf-deficit-side-type left-hemianopsia) (abneom-type hgaze-left)
        (facenumb-side left) (facial-side-type left-central)
        (weakness-type hemiparesis-left) (abndtrs-side left-incdtr)
        (babs-side left) (pp-side left-mild))
*d13*  ((right-frontal-lobe present) (right-parietal-lobe present)
        (right-temporal-lobe present))

*s14*  ((decloc-degree drowsy) (disoriented-degree mild)
        (hemineglect-side left) (denial present)
        (vf-deficit-side-type left-hemianopsia) (abneom-type hgaze-left)
        (facial-side-type left-central) (dysarthria-severity mild)
        (weakness-type hemiparesis-left) (abndtrs-side left-incdtr)
        (babs-side left) (dss-side left) (posloss-side left)
        (vibloss-side left) (twopoint-side left) (agraph-side left))
*d14*  ((right-frontal-lobe present) (right-parietal-lobe present)
        (right-temporal-lobe present))

*s15*  ((decloc-degree drowsy) (nystagmus-type vertical-upbeat)
        (abneom-type hgaze-left) (abneom-type vgaze-up)
        (abneom-type vgaze-down) (abneom-type ino-left)
        (facial-side-type right-central) (tongweak-side right)
        (dysarthria-severity mild) (weakness-type hemiparesis-right)
        (babs-side right))
*d15*  ((left-pons present))

*s16*  ((poorokn-direction lhoriz) (facial-side-type left-central)
        (dysarthria-severity mild) (weakness-type hemiparesis-left)
        (decram-side left-mild) (abndtrs-side left-incdtr))
*d16*  ((right-frontal-lobe present))

*s17*  ((facial-side-type right-central) (tongweak-side right)
        (dysarthria-severity moderate) (weakness-type hemiparesis-right)
        (abndtrs-side right-incdtr))
*d17*  ((left-frontal-lobe present))

*s18*  ((decloc-degree drowsy) (abneom-type hgaze-left) (facenumb-side left)
        (facial-side-type left-central) (weakness-type hemiparesis-left)
        (abndtrs-side left-decdtr) (babs-side left) (dss-side left)
        (pp-side left-mild) (posloss-side left) (vibloss-side left))
*d18*  ((right-frontal-lobe present) (right-parietal-lobe present))

*s19*  ((decloc-degree stupor) (vf-deficit-side-type right-hemianopsia)
        (abneom-type hgaze-right) (facenumb-side right)
        (facial-side-type right-central) (weakness-type hemiparesis-right)
        (abndtrs-side right-incdtr) (babs-side right)
        (pp-side right-moderate))
*d19*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s20*  ((poorokn-direction lhoriz) (facial-side-type left-central)
        (weakness-type hemiparesis-left) (babs-side left) (dss-side left))
```

```
*d20*  ((right-frontal-lobe present))

*s21*  ((disoriented-degree mild) (poorokn-direction rhoriz)
        (abneom-type hgaze-right) (facial-side-type right-central)
        (tongweak-side right) (weakness-type hemiparesis-right)
        (babs-side right))
*d21*  ((left-frontal-lobe present))

*s22*  ((disoriented-degree mild) (facenumb-side left)
        (facial-side-type left-central) (dysarthria-severity mild)
        (weakness-type hemiparesis-left) (babs-side left)
        (pp-side left-moderate) (posloss-side left) (vibloss-side left)
        (twopoint-side left) (agraph-side left))
*d22*  ((right-frontal-lobe present) (right-internal-capsule present)
        (right-thalamus present))

*s23*  ((decloc-degree drowsy) (disoriented-degree mild)
        (vf-deficit-side-type left-hemianopsia) (poorokn-direction lhoriz)
        (abneom-type hgaze-left) (facenumb-side left)
        (facial-side-type left-central) (weakness-type hemiparesis-left)
        (abndtrs-side left-decdtr) (dss-side left) (pp-side left-moderate)
        (posloss-side left) (agraph-side left))
*d23*  ((right-frontal-lobe present) (right-parietal-lobe present))

*s24*  ((compdef-severity severe) (nonfluency-severity severe)
        (repetition-severity severe) (anomia-severity severe)
        (facial-side-type right-central))
*d24*  ((left-frontal-lobe present) (left-temporal-lobe present))

*s25*  ((decloc-degree drowsy) (vf-deficit-side-type left-hemianopsia)
        (abneom-type hgaze-left) (facenumb-side left)
        (facial-side-type left-central) (dysarthria-severity moderate)
        (weakness-type hemiparesis-left) (dss-side left)
        (pp-side left-mild) (posloss-side left) (vibloss-side left)
        (twopoint-side left) (agraph-side left))
*d25*  ((right-frontal-lobe present) (right-parietal-lobe present)
        (right-temporal-lobe present))

*s26*  ((facial-side-type right-central) (dysarthria-severity moderate)
        (weakness-type hemiparesis-right) (babs-side right))
*d26*  ((left-internal-capsule present))

*s27*  ((disoriented-degree moderate) (compdef-severity severe)
        (nonfluency-severity severe) (repetition-severity severe)
        (anomia-severity severe) (vf-deficit-side-type right-hemianopsia)
        (poorokn-direction rhoriz) (facial-side-type right-central)
        (dysarthria-severity severe) (weakness-type hemiparesis-right)
        (decram-side right-severe) (abndtrs-side right-incdtr))
*d27*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s28*  ((facenumb-side right) (weakness-type hemiparesis-right)
        (ataxia-type limb-right-mild) (decram-side right-mild)
        (babs-side right) (gait-type unsteady))
```

```
*d28*  ((left-pons present))

*s29*  ((nonfluency-severity mild) (repetition-severity mild))
*d29*  ((left-frontal-lobe present))

*s30*  ((disoriented-degree mild) (nonfluency-severity mild)
        (repetition-severity mild) (anomia-severity mild)
        (weakness-type hemiparesis-right) (decram-side right-mild)
        (babs-side right) (vibloss-side right))
*d30*  ((left-internal-capsule present) (left-thalamus present))

*s31*  ((facial-side-type right-central) (dysarthria-severity mild)
        (weakness-type hemiparesis-right) (decram-side right-mild)
        (babs-side right))
*d31*  ((left-internal-capsule present))

*s32*  ((facial-side-type right-central) (dysarthria-severity moderate)
        (weakness-type hemiparesis-right) (babs-side right))
*d32*  ((left-internal-capsule present))

*s33*  ((decloc-degree drowsy) (denial present)
        (vf-deficit-side-type left-hemianopsia)
        (facial-side-type left-central) (dysarthria-severity moderate)
        (weakness-type hemiparesis-left) (decram-side left-mild)
        (abndtrs-side left-incdtr) (babs-side left))
*d33*  ((right-frontal-lobe present) (right-parietal-lobe present))

*s34*  ((abneom-type fourn-right) (ataxia-type limb-left-mild)
        (decram-side left-mild) (babs-side left))
*d34*  ((right-midbrain present))

*s35*  ((dyspraxia present) (compdef-severity severe)
        (nonfluency-severity severe) (repetition-severity severe)
        (anomia-severity severe) (vf-deficit-side-type right-hemianopsia)
        (poorokn-direction rhoriz) (facial-side-type right-central)
        (dysarthria-severity severe) (weakness-type hemiparesis-right)
        (abndtrs-side right-incdtr) (babs-side right))
*d35*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s36*  ((decloc-degree drowsy) (disoriented-degree severe)
        (compdef-severity severe) (nonfluency-severity severe)
        (repetition-severity severe) (anomia-severity severe)
        (vf-deficit-side-type right-hemianopsia)
        (facial-side-type right-central)
        (dysarthria-severity mild) (weakness-type hemiparesis-right))
*d36*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s37*  ((facenumb-side right) (facial-side-type right-central)
        (dysarthria-severity mild) (weakness-type hemiparesis-right)
        (ataxia-type limb-left-mild) (decram-side left-mild)
        (babs-side right) (pp-side right-mild) (posloss-side right))
*d37*  ((left-pons present))
```

```
*s38*  ((disoriented-degree moderate) (hemineglect-side right)
        (denial present) (compdef-severity moderate)
        (nonfluency-severity moderate) (repetition-severity moderate)
        (anomia-severity mild) (vf-deficit-side-type right-hemianopsia)
        (poorokn-direction rhoriz) (abneom-type hgaze-right)
        (facial-side-type right-central) (weakness-type hemiparesis-right)
        (abndtrs-side right-incdtr) (babs-side right) (dss-side right)
        (pp-side right-moderate) (vibloss-side right) (agraph-side right))
*d38*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s39*  ((decloc-degree drowsy) (disoriented-degree moderate)
        (denial present) (vf-deficit-side-type left-hemianopsia)
        (abneom-type hgaze-left) (facenumb-side left)
        (facial-side-type left-central) (dysarthria-severity mild)
        (weakness-type hemiparesis-left) (abndtrs-side left-incdtr)
        (babs-side left) (dss-side left) (pp-side left-moderate))
*d39*  ((right-frontal-lobe present) (right-parietal-lobe present)
        (right-temporal-lobe present))

*s40* ((facial-side-type right-central) (dysarthria-severity mild)
        (weakness-type hemiparesis-right) (babs-side right))
*d40*  ((left-internal-capsule present))

*s41*  ((poorokn-direction lhoriz) (facial-side-type left-central)
        (weakness-type hemiparesis-left) (decram-side left-severe))
*d41*  ((right-internal-capsule present))

*s42*  ((decloc-degree drowsy) (compdef-severity severe)
        (nonfluency-severity severe) (repetition-severity severe)
        (anomia-severity severe) (vf-deficit-side-type right-hemianopsia)
        (abneom-type hgaze-right) (facenumb-side right)
        (facial-side-type right-central) (weakness-type hemiparesis-right)
        (abndtrs-side right-decdtr) (pp-side right-severe))
*d42*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s43*  ((decloc-degree drowsy) (hemineglect-side right)
        (compdef-severity severe) (nonfluency-severity severe)
        (repetition-severity severe) (anomia-severity severe)
        (vf-deficit-side-type right-hemianopsia) (abneom-type hgaze-right)
        (facenumb-side right) (facial-side-type right-central)
        (dysarthria-severity severe) (weakness-type hemiparesis-right)
        (abndtrs-side right-incdtr) (babs-side right) (pp-side right-mild))
*d43*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s44*  ((facial-side-type left-central) (weakness-type hemiparesis-left)
        (decram-side left-mild))
*d44*  ((right-internal-capsule present))

*s45*  ((compdef-severity severe) (nonfluency-severity severe)
        (repetition-severity severe) (anomia-severity severe)
        (facial-side-type right-central) (weakness-type hemiparesis-right))
*d45*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s46*  ((dyspraxia present) (hemineglect-side right) (compdef-severity mild)
        (nonfluency-severity severe) (repetition-severity severe)
        (anomia-severity severe) (vf-deficit-side-type right-hemianopsia)
        (poorokn-direction rhoriz) (abneom-type hgaze-right)
        (facenumb-side right) (facial-side-type right-central)
        (dysarthria-severity moderate) (weakness-type hemiparesis-right)
        (abndtrs-side right-incdtr) (babs-side right)
        (dss-side right) (pp-side right-mild))
*d46*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s47*  ((decloc-degree drowsy) (compdef-severity severe)
        (nonfluency-severity severe) (repetition-severity severe)
        (anomia-severity severe) (vf-deficit-side-type right-hemianopsia)
        (abneom-type hgaze-right) (facenumb-side right)
        (facial-side-type right-central) (dysarthria-severity severe)
        (weakness-type hemiparesis-right) (abndtrs-side right-incdtr)
        (babs-side right) (pp-side right-moderate))
*d47*  ((left-frontal-lobe present) (left-parietal-lobe present)
        (left-temporal-lobe present))

*s48*  ((ataxia-type limb-right-mild) (decram-side right-mild)
        (gait-type unsteady))
*d48*  ((right-cerebellar-hemisphere present))

*s49*  ((compdef-severity mild) (nonfluency-severity mild)
        (repetition-severity mild) (dysarthria-severity mild)
        (weakness-type hemiparesis-left) (decram-side left-mild)
        (dss-side left) (pp-side left-mild))
*d49*  ((right-frontal-lobe present) (right-parietal-lobe present))

*s50*  ((nonfluency-severity mild) (facenumb-side right)
        (facial-side-type right-central) (dysarthria-severity moderate)
        (weakness-type hemiparesis-right) (ataxia-type limb-right-mild)
        (decram-side right-mild) (abndtrs-side right-incdtr)
        (babs-side right) (pp-side right-mild))
*d50*  ((left-frontal-lobe present) (left-parietal-lobe present))
```

Appendix C

Model-Based Diagnosis

C.1 Full Adder

C.1.1 Knowledge Base

```
(setf *brules* '(
((out ?x ?w ?t) <- (andg ?x) (in1 ?x ?u ?t) (in2 ?x ?v ?t)
                   (norm ?x) (and ?u ?v ?w))
((out ?x ?w ?t) <- (org ?x) (in1 ?x ?u ?t) (in2 ?x ?v ?t)
                   (norm ?x) (or ?u ?v ?w))
((out ?x ?w ?t) <- (xorg ?x) (in1 ?x ?u ?t) (in2 ?x ?v ?t)
                   (norm ?x) (xor ?u ?v ?w))

((out ?x ?w ?t) <- (in1 ?x ?u ?t) (in2 ?x ?v ?t) (ab ?x ?u ?v ?w ?t))
((out ?x 0 ?t) <- (in1 ?x ?u ?t) (in2 ?x ?v ?t) (stuck-at-0 ?x))
((out ?x 1 ?t) <- (in1 ?x ?u ?t) (in2 ?x ?v ?t) (stuck-at-1 ?x))

; brules specifically for full adder

((in1 a1 ?y ?t) <- (in1 x1 ?y ?t))
((in2 a1 ?y ?t) <- (in2 x1 ?y ?t))
((in1 x2 ?y ?t) <- (out x1 ?y ?t))
((in2 x2 ?y ?t) <- (in1 a2 ?y ?t))
((in2 a2 ?y ?t) <- (out x1 ?y ?t))
((in1 o1 ?y ?t) <- (out a2 ?y ?t))
((in2 o1 ?y ?t) <- (out a1 ?y ?t))
((in1 x1 ?u ?t) <- (given-in1 x1 ?u ?t))
((in2 x1 ?u ?t) <- (given-in2 x1 ?u ?t))
((in1 a2 ?u ?t) <- (given-in1 a2 ?u ?t))
))

(setf *facts*
      '((and 0 0 0) (and 0 1 0) (and 1 0 0) (and 1 1 1)
        (or 0 0 0) (or 0 1 1) (or 1 0 1) (or 1 1 1)
        (xor 0 0 0) (xor 0 1 1) (xor 1 0 1) (xor 1 1 0)

        ; facts specifically for full adder

        (xorg x1) (xorg x2) (andg a1) (andg a2) (org o1)))

(setf *nogoods* nil)
(setf *assumption-nogoods* nil)

(setf *inter-batch-beam-width* 45)
```

```
(setf *intra-batch-beam-width* 45)
(setf *bchain-depth* 7)
(setf *caching* t)
(setf *factoring* nil)
(setf *remove-superset?* nil)
(setf *explanation-eval-metric* #'diag-simplicity)
(setf *compute-estimate-fn* #'diag-compute-estimate)
(setf *combine-estimates-fn* #'diag-combine-estimates)

(setf *predicate-specific-abduction* t)
(setf *assumable-predicates*
      '(norm ab stuck-at-0 stuck-at-1 given-in1 given-in2))
(setf *free-assumption-predicates*
      '(norm given-in1 given-in2))
(setf *fault-mode-predicates*
      '(stuck-at-0 stuck-at-1))
(setf *components* '(x1 x2 a1 a2 o1))
```

C.1.2 Test Data

For each of the 10 scenarios tested, we give: (1) the faults present; and (2) the input atoms representing the dynamic behavior.

```
#1: ((stuck-at-0 a1) (stuck-at-0 a2))

(((in1 x1 0 t4) (in2 x1 1 t4) (in1 a2 1 t4) (out x2 0 t4)
  (out o1 0 t4) (out x1 1 t4) (out a1 0 t4) (out a2 0 t4))
 ((in1 x1 1 t6) (in2 x1 0 t6) (in1 a2 1 t6) (out x2 0 t6)
  (out o1 0 t6) (out x1 1 t6) (out a1 0 t6) (out a2 0 t6))
 ((in1 x1 1 t7) (in2 x1 1 t7) (in1 a2 0 t7) (out x2 0 t7)
  (out o1 0 t7) (out x1 0 t7) (out a1 0 t7) (out a2 0 t7))
 ((in1 x1 1 t8) (in2 x1 1 t8) (in1 a2 1 t8) (out x2 1 t8)
  (out o1 0 t8) (out x1 0 t8) (out a1 0 t8) (out a2 0 t8)))

#2: ((stuck-at-1 x1) (stuck-at-1 a1))

(((in1 x1 0 t1) (in2 x1 0 t1) (in1 a2 0 t1) (out x2 1 t1)
  (out o1 1 t1) (out x1 1 t1) (out a1 1 t1) (out a2 0 t1))
 ((in1 x1 0 t2) (in2 x1 0 t2) (in1 a2 1 t2) (out x2 0 t2)
  (out o1 1 t2) (out x1 1 t2) (out a1 1 t2) (out a2 1 t2))
 ((in1 x1 0 t3) (in2 x1 1 t3) (in1 a2 0 t3) (out x2 1 t3)
  (out o1 1 t3) (out x1 1 t3) (out a1 1 t3) (out a2 0 t3))
 ((in1 x1 1 t5) (in2 x1 0 t5) (in1 a2 0 t5) (out x2 1 t5)
  (out o1 1 t5) (out x1 1 t5) (out a1 1 t5) (out a2 0 t5))
 ((in1 x1 1 t7) (in2 x1 1 t7) (in1 a2 0 t7) (out x2 1 t7)
  (out o1 1 t7) (out x1 1 t7) (out a1 1 t7) (out a2 0 t7))
 ((in1 x1 1 t8) (in2 x1 1 t8) (in1 a2 1 t8) (out x2 0 t8)
  (out o1 1 t8) (out x1 1 t8) (out a1 1 t8) (out a2 1 t8)))

#3: ((stuck-at-0 x1) (stuck-at-1 o1))

(((in1 x1 0 t1) (in2 x1 0 t1) (in1 a2 0 t1) (out x2 0 t1)
  (out o1 1 t1) (out x1 0 t1) (out a1 0 t1) (out a2 0 t1))
 ((in1 x1 0 t2) (in2 x1 0 t2) (in1 a2 1 t2) (out x2 1 t2)
  (out o1 1 t2) (out x1 0 t2) (out a1 0 t2) (out a2 0 t2))
 ((in1 x1 0 t3) (in2 x1 1 t3) (in1 a2 0 t3) (out x2 0 t3)
  (out o1 1 t3) (out x1 0 t3) (out a1 0 t3) (out a2 0 t3))
 ((in1 x1 0 t4) (in2 x1 1 t4) (in1 a2 1 t4) (out x2 1 t4)
  (out o1 1 t4) (out x1 0 t4) (out a1 0 t4) (out a2 0 t4))
 ((in1 x1 1 t5) (in2 x1 0 t5) (in1 a2 0 t5) (out x2 0 t5)
  (out o1 1 t5) (out x1 0 t5) (out a1 0 t5) (out a2 0 t5))
 ((in1 x1 1 t6) (in2 x1 0 t6) (in1 a2 1 t6) (out x2 1 t6)
  (out o1 1 t6) (out x1 0 t6) (out a1 0 t6) (out a2 0 t6)))

#4: ((stuck-at-1 x1) (stuck-at-1 a1))

(((in1 x1 0 t1) (in2 x1 0 t1) (in1 a2 0 t1) (out x2 1 t1)
  (out o1 1 t1) (out x1 1 t1) (out a1 1 t1) (out a2 0 t1))
 ((in1 x1 0 t2) (in2 x1 0 t2) (in1 a2 1 t2) (out x2 0 t2)
  (out o1 1 t2) (out x1 1 t2) (out a1 1 t2) (out a2 1 t2))
 ((in1 x1 0 t3) (in2 x1 1 t3) (in1 a2 0 t3) (out x2 1 t3)
  (out o1 1 t3) (out x1 1 t3) (out a1 1 t3) (out a2 0 t3))
 ((in1 x1 1 t5) (in2 x1 0 t5) (in1 a2 0 t5) (out x2 1 t5)
  (out o1 1 t5) (out x1 1 t5) (out a1 1 t5) (out a2 0 t5))
 ((in1 x1 1 t7) (in2 x1 1 t7) (in1 a2 0 t7) (out x2 1 t7)
  (out o1 1 t7) (out x1 1 t7) (out a1 1 t7) (out a2 0 t7))
 ((in1 x1 1 t8) (in2 x1 1 t8) (in1 a2 1 t8) (out x2 0 t8)
  (out o1 1 t8) (out x1 1 t8) (out a1 1 t8) (out a2 1 t8)))

#5: ((stuck-at-0 o1))

(((in1 x1 0 t4) (in2 x1 1 t4) (in1 a2 1 t4) (out x2 0 t4)
  (out o1 0 t4) (out x1 1 t4) (out a1 0 t4) (out a2 1 t4))
 ((in1 x1 1 t6) (in2 x1 0 t6) (in1 a2 1 t6) (out x2 0 t6)
  (out o1 0 t6) (out x1 1 t6) (out a1 0 t6) (out a2 1 t6))
 ((in1 x1 1 t7) (in2 x1 1 t7) (in1 a2 0 t7) (out x2 0 t7)
  (out o1 0 t7) (out x1 0 t7) (out a1 1 t7) (out a2 0 t7))
 ((in1 x1 1 t8) (in2 x1 1 t8) (in1 a2 1 t8) (out x2 1 t8)
  (out o1 0 t8) (out x1 0 t8) (out a1 1 t8) (out a2 0 t8)))

#6: ((stuck-at-1 x1) (stuck-at-0 a2))

(((in1 x1 0 t1) (in2 x1 0 t1) (in1 a2 0 t1) (out x2 1 t1)
  (out o1 0 t1) (out x1 1 t1) (out a1 0 t1) (out a2 0 t1))
 ((in1 x1 0 t2) (in2 x1 0 t2) (in1 a2 1 t2) (out x2 0 t2)
  (out o1 0 t2) (out x1 1 t2) (out a1 0 t2) (out a2 0 t2))
 ((in1 x1 0 t4) (in2 x1 1 t4) (in1 a2 1 t4) (out x2 0 t4)
  (out o1 0 t4) (out x1 1 t4) (out a1 0 t4) (out a2 0 t4))
 ((in1 x1 1 t6) (in2 x1 0 t6) (in1 a2 1 t6) (out x2 0 t6)
  (out o1 0 t6) (out x1 1 t6) (out a1 0 t6) (out a2 0 t6))
 ((in1 x1 1 t7) (in2 x1 1 t7) (in1 a2 0 t7) (out x2 1 t7)
  (out o1 1 t7) (out x1 1 t7) (out a1 1 t7) (out a2 0 t7))
 ((in1 x1 1 t8) (in2 x1 1 t8) (in1 a2 1 t8) (out x2 0 t8)
  (out o1 1 t8) (out x1 1 t8) (out a1 1 t8) (out a2 0 t8)))

#7: ((ab a1) (stuck-at-1 a2))
```

```
(((in1 x1 0 t1) (in2 x1 0 t1) (in1 a2 0 t1) (out x2 0 t1)
  (out o1 1 t1) (out x1 0 t1) (out a1 1 t1) (out a2 1 t1))
 ((in1 x1 0 t2) (in2 x1 0 t2) (in1 a2 1 t2) (out x2 1 t2)
  (out o1 1 t2) (out x1 0 t2) (out a1 1 t2) (out a2 1 t2))
 ((in1 x1 0 t3) (in2 x1 1 t3) (in1 a2 0 t3) (out x2 1 t3)
  (out o1 1 t3) (out x1 1 t3) (out a1 1 t3) (out a2 1 t3))
 ((in1 x1 1 t5) (in2 x1 0 t5) (in1 a2 0 t5) (out x2 1 t5)
  (out o1 1 t5) (out x1 1 t5) (out a1 0 t5) (out a2 1 t5)))
```

```
#8: ((stuck-at-1 a1) (ab o1))
```

```
(((in1 x1 0 t3) (in2 x1 1 t3) (in1 a2 0 t3) (out x2 1 t3)
  (out o1 1 t3) (out x1 1 t3) (out a1 1 t3) (out a2 0 t3))
 ((in1 x1 1 t5) (in2 x1 0 t5) (in1 a2 0 t5) (out x2 1 t5)
  (out o1 1 t5) (out x1 1 t5) (out a1 1 t5) (out a2 0 t5))
 ((in1 x1 1 t7) (in2 x1 1 t7) (in1 a2 0 t7) (out x2 0 t7)
  (out o1 0 t7) (out x1 0 t7) (out a1 1 t7) (out a2 0 t7)))
```

```
#9: ((stuck-at-1 x2) (stuck-at-0 o1))
```

```
(((in1 x1 0 t1) (in2 x1 0 t1) (in1 a2 0 t1) (out x2 1 t1)
  (out o1 0 t1) (out x1 0 t1) (out a1 0 t1) (out a2 0 t1))
 ((in1 x1 0 t4) (in2 x1 1 t4) (in1 a2 1 t4) (out x2 1 t4)
  (out o1 0 t4) (out x1 1 t4) (out a1 0 t4) (out a2 1 t4))
 ((in1 x1 1 t6) (in2 x1 0 t6) (in1 a2 1 t6) (out x2 1 t6)
  (out o1 0 t6) (out x1 1 t6) (out a1 0 t6) (out a2 1 t6))
 ((in1 x1 1 t7) (in2 x1 1 t7) (in1 a2 0 t7) (out x2 1 t7)
  (out o1 0 t7) (out x1 0 t7) (out a1 1 t7) (out a2 0 t7))
 ((in1 x1 1 t8) (in2 x1 1 t8) (in1 a2 1 t8) (out x2 1 t8)
  (out o1 0 t8) (out x1 0 t8) (out a1 1 t8) (out a2 0 t8)))
```

```
#10: ((stuck-at-0 a2))
```

```
(((in1 x1 0 t4) (in2 x1 1 t4) (in1 a2 1 t4) (out x2 0 t4)
  (out o1 0 t4) (out x1 1 t4) (out a1 0 t4) (out a2 0 t4))
 ((in1 x1 1 t6) (in2 x1 0 t6) (in1 a2 1 t6) (out x2 0 t6)
  (out o1 0 t6) (out x1 1 t6) (out a1 0 t6) (out a2 0 t6)))
```

C.2 Dynamic Devices

C.2.1 Knowledge Base of General QSIM Constraints

```
(setf *brules* '(
((holds.m0+ ?f ?g ?m1 inc ?m2 inc)
 <- (pos ?m1) (pos ?m2) (corr-mag.m0+ ?f ?g ?m1 ?m2))
((holds.m0+ ?f ?g ?m1 std ?m2 std)
 <- (pos ?m1) (pos ?m2) (corr-mag.m0+ ?f ?g ?m1 ?m2))
((holds.m0+ ?f ?g ?m1 dec ?m2 dec)
 <- (pos ?m1) (pos ?m2) (corr-mag.m0+ ?f ?g ?m1 ?m2))

((holds.m0+ ?f ?g ?m1 inc ?m2 inc)
 <- (neg ?m1) (neg ?m2) (corr-mag.m0+ ?f ?g ?m1 ?m2))
((holds.m0+ ?f ?g ?m1 std ?m2 std)
 <- (neg ?m1) (neg ?m2) (corr-mag.m0+ ?f ?g ?m1 ?m2))
((holds.m0+ ?f ?g ?m1 dec ?m2 dec)
 <- (neg ?m1) (neg ?m2) (corr-mag.m0+ ?f ?g ?m1 ?m2))

((holds.- ?f ?g ?h ?m inc ?m inc 0 inc)
 <- (corr-val.- ?f ?g ?h ?m inc ?m inc 0 inc))
((holds.- ?f ?g ?h ?m inc ?m inc 0 std)
 <- (corr-val.- ?f ?g ?h ?m inc ?m inc 0 std))
((holds.- ?f ?g ?h ?m inc ?m inc 0 dec)
 <- (corr-val.- ?f ?g ?h ?m inc ?m inc 0 dec))
((holds.- ?f ?g ?h ?m dec ?m dec 0 inc)
 <- (corr-val.- ?f ?g ?h ?m dec ?m dec 0 inc))
((holds.- ?f ?g ?h ?m dec ?m dec 0 std)
 <- (corr-val.- ?f ?g ?h ?m dec ?m dec 0 std))
((holds.- ?f ?g ?h ?m dec ?m dec 0 dec)
 <- (corr-val.- ?f ?g ?h ?m dec ?m dec 0 dec))

((holds.- ?f ?g ?h ?m inc 0 inc ?m inc)
 <- (corr-val.- ?f ?g ?h ?m inc 0 inc ?m inc))
((holds.- ?f ?g ?h ?m inc 0 inc ?m std)
 <- (corr-val.- ?f ?g ?h ?m inc 0 inc ?m std))
((holds.- ?f ?g ?h ?m inc 0 inc ?m dec)
 <- (corr-val.- ?f ?g ?h ?m inc 0 inc ?m dec))
((holds.- ?f ?g ?h ?m dec 0 dec ?m inc)
 <- (corr-val.- ?f ?g ?h ?m dec 0 dec ?m inc))
((holds.- ?f ?g ?h ?m dec 0 dec ?m std)
 <- (corr-val.- ?f ?g ?h ?m dec 0 dec ?m std))
((holds.- ?f ?g ?h ?m dec 0 dec ?m dec)
 <- (corr-val.- ?f ?g ?h ?m dec 0 dec ?m dec))

((holds.- ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-val.- ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc))
((holds.- ?f ?g ?h ?m1 inc ?m2 inc ?m3 std)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-val.- ?f ?g ?h ?m1 inc ?m2 inc ?m3 std))
((holds.- ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-val.- ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec))
((holds.- ?f ?g ?h ?m1 inc ?m2 std ?m3 inc)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-mag.- ?f ?g ?h ?m1 ?m2 ?m3))
((holds.- ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-mag.- ?f ?g ?h ?m1 ?m2 ?m3))
((holds.- ?f ?g ?h ?m1 std ?m2 inc ?m3 dec)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-mag.- ?f ?g ?h ?m1 ?m2 ?m3))
((holds.- ?f ?g ?h ?m1 std ?m2 std ?m3 std)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-mag.- ?f ?g ?h ?m1 ?m2 ?m3))
((holds.- ?f ?g ?h ?m1 std ?m2 dec ?m3 inc)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-mag.- ?f ?g ?h ?m1 ?m2 ?m3))
```

```
((holds.- ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-mag.- ?f ?g ?h ?m1 ?m2 ?m3))
((holds.- ?f ?g ?h ?m1 dec ?m2 std ?m3 dec)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-mag.- ?f ?g ?h ?m1 ?m2 ?m3))
((holds.- ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-val.- ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc))
((holds.- ?f ?g ?h ?m1 dec ?m2 dec ?m3 std)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-val.- ?f ?g ?h ?m1 dec ?m2 dec ?m3 std))
((holds.- ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec)
 <- (<> ?m1 ?m2) (<> ?m1 ?m3)
    (corr-val.- ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec))

((holds.* ?f ?g ?h 0 inc ?m inc 0 inc) <- (pos ?m))
((holds.* ?f ?g ?h 0 inc ?m std 0 inc) <- (pos ?m))
((holds.* ?f ?g ?h 0 inc ?m dec 0 inc) <- (pos ?m))
((holds.* ?f ?g ?h 0 std ?m inc 0 std) <- (pos ?m))
((holds.* ?f ?g ?h 0 std ?m std 0 std) <- (pos ?m))
((holds.* ?f ?g ?h 0 std ?m dec 0 std) <- (pos ?m))
((holds.* ?f ?g ?h 0 dec ?m inc 0 dec) <- (pos ?m))
((holds.* ?f ?g ?h 0 dec ?m std 0 dec) <- (pos ?m))
((holds.* ?f ?g ?h 0 dec ?m dec 0 dec) <- (pos ?m))

((holds.* ?f ?g ?h 0 inc ?m inc 0 dec) <- (neg ?m))
((holds.* ?f ?g ?h 0 inc ?m std 0 dec) <- (neg ?m))
((holds.* ?f ?g ?h 0 inc ?m dec 0 dec) <- (neg ?m))
((holds.* ?f ?g ?h 0 std ?m inc 0 std) <- (neg ?m))
((holds.* ?f ?g ?h 0 std ?m std 0 std) <- (neg ?m))
((holds.* ?f ?g ?h 0 std ?m dec 0 std) <- (neg ?m))
((holds.* ?f ?g ?h 0 dec ?m inc 0 inc) <- (neg ?m))
((holds.* ?f ?g ?h 0 dec ?m std 0 inc) <- (neg ?m))
((holds.* ?f ?g ?h 0 dec ?m dec 0 inc) <- (neg ?m))

((holds.* ?f ?g ?h ?m inc 0 inc 0 inc) <- (pos ?m))
((holds.* ?f ?g ?h ?m inc 0 std 0 std) <- (pos ?m))
((holds.* ?f ?g ?h ?m inc 0 dec 0 dec) <- (pos ?m))
((holds.* ?f ?g ?h ?m std 0 inc 0 inc) <- (pos ?m))
((holds.* ?f ?g ?h ?m std 0 std 0 std) <- (pos ?m))
((holds.* ?f ?g ?h ?m std 0 dec 0 dec) <- (pos ?m))
((holds.* ?f ?g ?h ?m dec 0 inc 0 inc) <- (pos ?m))
((holds.* ?f ?g ?h ?m dec 0 std 0 std) <- (pos ?m))
((holds.* ?f ?g ?h ?m dec 0 dec 0 dec) <- (pos ?m))

((holds.* ?f ?g ?h ?m inc 0 inc 0 dec) <- (neg ?m))
((holds.* ?f ?g ?h ?m inc 0 std 0 std) <- (neg ?m))
((holds.* ?f ?g ?h ?m inc 0 dec 0 inc) <- (neg ?m))
((holds.* ?f ?g ?h ?m std 0 inc 0 dec) <- (neg ?m))
((holds.* ?f ?g ?h ?m std 0 std 0 std) <- (neg ?m))
((holds.* ?f ?g ?h ?m std 0 dec 0 inc) <- (neg ?m))
((holds.* ?f ?g ?h ?m dec 0 inc 0 dec) <- (neg ?m))
((holds.* ?f ?g ?h ?m dec 0 std 0 std) <- (neg ?m))
```

```
((holds.* ?f ?g ?h ?m dec 0 dec 0 inc) <- (neg ?m))

((holds.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 inc ?m2 std ?m3 inc)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc))
((holds.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 std)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 std))
((holds.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 dec)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 dec))
((holds.* ?f ?g ?h ?m1 std ?m2 inc ?m3 inc)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 std ?m2 std ?m3 std)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 std ?m2 dec ?m3 dec)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 inc)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 inc))
((holds.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 std)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 std))
((holds.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec))
((holds.* ?f ?g ?h ?m1 dec ?m2 std ?m3 dec)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))

((holds.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc))
((holds.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 std)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 std))
((holds.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec))
((holds.* ?f ?g ?h ?m1 inc ?m2 std ?m3 dec)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 dec)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 std ?m2 inc ?m3 inc)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 std ?m2 std ?m3 std)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 std ?m2 dec ?m3 dec)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
```

```
((holds.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 inc)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 dec ?m2 std ?m3 inc)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc))
((holds.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 std)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 std))
((holds.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec))

((holds.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc))
((holds.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 std)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 std))
((holds.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec))
((holds.* ?f ?g ?h ?m1 inc ?m2 std ?m3 inc)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 std ?m2 inc ?m3 dec)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 std ?m2 std ?m3 std)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 std ?m2 dec ?m3 inc)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 dec ?m2 std ?m3 dec)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc))
((holds.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 std)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 std))
((holds.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec))

((holds.* ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 inc ?m2 std ?m3 dec)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc))
```

```
((holds.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 std)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 std))
((holds.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 dec)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 inc ?m2 dec ?m3 dec))
((holds.* ?f ?g ?h ?m1 std ?m2 inc ?m3 dec)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 std ?m2 std ?m3 std)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 std ?m2 dec ?m3 inc)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 inc)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 inc))
((holds.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 std)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 std))
((holds.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val.* ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec))
((holds.* ?f ?g ?h ?m1 dec ?m2 std ?m3 inc)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))
((holds.* ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag.* ?f ?g ?h ?m1 ?m2 ?m3))

((holds./ ?f ?g ?h 0 inc ?m inc 0 inc) <- (pos ?m))
((holds./ ?f ?g ?h 0 inc ?m std 0 inc) <- (pos ?m))
((holds./ ?f ?g ?h 0 inc ?m dec 0 inc) <- (pos ?m))
((holds./ ?f ?g ?h 0 std ?m inc 0 std) <- (pos ?m))
((holds./ ?f ?g ?h 0 std ?m std 0 std) <- (pos ?m))
((holds./ ?f ?g ?h 0 std ?m dec 0 std) <- (pos ?m))
((holds./ ?f ?g ?h 0 dec ?m inc 0 dec) <- (pos ?m))
((holds./ ?f ?g ?h 0 dec ?m std 0 dec) <- (pos ?m))
((holds./ ?f ?g ?h 0 dec ?m dec 0 dec) <- (pos ?m))

((holds./ ?f ?g ?h 0 inc ?m inc 0 dec) <- (neg ?m))
((holds./ ?f ?g ?h 0 inc ?m std 0 dec) <- (neg ?m))
((holds./ ?f ?g ?h 0 inc ?m dec 0 dec) <- (neg ?m))
((holds./ ?f ?g ?h 0 std ?m inc 0 std) <- (neg ?m))
((holds./ ?f ?g ?h 0 std ?m std 0 std) <- (neg ?m))
((holds./ ?f ?g ?h 0 std ?m dec 0 std) <- (neg ?m))
((holds./ ?f ?g ?h 0 dec ?m inc 0 inc) <- (neg ?m))
((holds./ ?f ?g ?h 0 dec ?m std 0 inc) <- (neg ?m))
((holds./ ?f ?g ?h 0 dec ?m dec 0 inc) <- (neg ?m))

((holds./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc))
((holds./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 std)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 std))
((holds./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
```

```
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec))
((holds./ ?f ?g ?h ?m1 inc ?m2 std ?m3 inc)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 std ?m2 inc ?m3 dec)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 std ?m2 std ?m3 std)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 std ?m2 dec ?m3 inc)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 dec ?m2 std ?m3 dec)
 <- (pos ?m1) (pos ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc))
((holds./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 std)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 std))
((holds./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec)
 <- (pos ?m1) (pos ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec))

((holds./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 inc ?m2 std ?m3 dec)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc))
((holds./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 std)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 std))
((holds./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 dec)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 dec))
((holds./ ?f ?g ?h ?m1 std ?m2 inc ?m3 dec)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 std ?m2 std ?m3 std)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 std ?m2 dec ?m3 inc)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 inc)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 inc))
((holds./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 std)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 std))
((holds./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec)
 <- (pos ?m1) (neg ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec))
((holds./ ?f ?g ?h ?m1 dec ?m2 std ?m3 inc)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc)
 <- (pos ?m1) (neg ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))

((holds./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 inc ?m2 std ?m3 inc)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 inc))
((holds./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 std)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 std))
((holds./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 dec)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 dec))
((holds./ ?f ?g ?h ?m1 std ?m2 inc ?m3 inc)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 std ?m2 std ?m3 std)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 std ?m2 dec ?m3 dec)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 inc)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 inc))
((holds./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 std)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 std))
((holds./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec)
 <- (neg ?m1) (pos ?m2) (neg ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 dec))
((holds./ ?f ?g ?h ?m1 dec ?m2 std ?m3 dec)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec)
 <- (neg ?m1) (pos ?m2) (neg ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))

((holds./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 inc))
((holds./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 std)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 std))
((holds./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 inc ?m2 inc ?m3 dec))
((holds./ ?f ?g ?h ?m1 inc ?m2 std ?m3 dec)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 inc ?m2 dec ?m3 dec)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 std ?m2 inc ?m3 inc)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 std ?m2 std ?m3 std)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
```

```
((holds./ ?f ?g ?h ?m1 std ?m2 dec ?m3 dec)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 dec ?m2 inc ?m3 inc)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 dec ?m2 std ?m3 inc)
 <- (neg ?m1) (neg ?m2) (pos ?m3) (corr-mag./ ?f ?g ?h ?m1 ?m2 ?m3))
((holds./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 inc))
((holds./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 std)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 std))
((holds./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec)
 <- (neg ?m1) (neg ?m2) (pos ?m3)
    (corr-val./ ?f ?g ?h ?m1 dec ?m2 dec ?m3 dec))

((holds.d/dt ?f ?g ?m1 inc ?m2 inc)
 <- (pos ?m2) (corr-val.d/dt ?f ?g ?m1 inc ?m2 inc))
((holds.d/dt ?f ?g ?m1 inc ?m2 std)
 <- (pos ?m2) (corr-val.d/dt ?f ?g ?m1 inc ?m2 std))
((holds.d/dt ?f ?g ?m1 inc ?m2 dec)
 <- (pos ?m2) (corr-val.d/dt ?f ?g ?m1 inc ?m2 dec))
((holds.d/dt ?f ?g ?m1 std 0 inc)
 <- (corr-val.d/dt ?f ?g ?m1 std 0 inc))
((holds.d/dt ?f ?g ?m1 std 0 std)
 <- (corr-val.d/dt ?f ?g ?m1 std 0 std))
((holds.d/dt ?f ?g ?m1 std 0 dec)
 <- (corr-val.d/dt ?f ?g ?m1 std 0 dec))
((holds.d/dt ?f ?g ?m1 dec ?m2 inc)
 <- (neg ?m2) (corr-val.d/dt ?f ?g ?m1 dec ?m2 inc))
((holds.d/dt ?f ?g ?m1 dec ?m2 std)
 <- (neg ?m2) (corr-val.d/dt ?f ?g ?m1 dec ?m2 std))
((holds.d/dt ?f ?g ?m1 dec ?m2 dec)
 <- (neg ?m2) (corr-val.d/dt ?f ?g ?m1 dec ?m2 dec))
))

(setf *facts* '(
(holds.m0+ ?f ?g 0 inc 0 inc) (holds.m0+ ?f ?g 0 std 0 std)
(holds.m0+ ?f ?g 0 dec 0 dec)

(holds.- ?f ?g ?h ?m inc ?m std 0 inc)
(holds.- ?f ?g ?h ?m inc ?m dec 0 inc)
(holds.- ?f ?g ?h ?m std ?m inc 0 dec)
(holds.- ?f ?g ?h ?m std ?m std 0 std)
(holds.- ?f ?g ?h ?m std ?m dec 0 inc)
(holds.- ?f ?g ?h ?m dec ?m inc 0 dec)
(holds.- ?f ?g ?h ?m dec ?m std 0 dec)

(holds.- ?f ?g ?h ?m inc 0 std ?m inc)
(holds.- ?f ?g ?h ?m inc 0 dec ?m inc)
(holds.- ?f ?g ?h ?m std 0 inc ?m dec)
(holds.- ?f ?g ?h ?m std 0 std ?m std)
(holds.- ?f ?g ?h ?m std 0 dec ?m inc)
(holds.- ?f ?g ?h ?m dec 0 inc ?m dec)
```

```
(holds.- ?f ?g ?h ?m dec 0 std ?m dec)

(holds.* ?f ?g ?h 0 inc 0 inc 0 std)
(holds.* ?f ?g ?h 0 inc 0 std 0 std)
(holds.* ?f ?g ?h 0 inc 0 dec 0 std)
(holds.* ?f ?g ?h 0 std 0 inc 0 std)
(holds.* ?f ?g ?h 0 std 0 std 0 std)
(holds.* ?f ?g ?h 0 std 0 dec 0 std)
(holds.* ?f ?g ?h 0 dec 0 inc 0 std)
(holds.* ?f ?g ?h 0 dec 0 std 0 std)
(holds.* ?f ?g ?h 0 dec 0 dec 0 std)
))

(setf *nogoods* '(
((pos 0)) ((neg 0)) ((pos ?m) (neg ?m))

((corr-val.- ?f ?g ?h ?m1 ?d1 ?m2 ?d2 0 ?d3) (<> ?m1 ?m2))
((corr-mag.- ?f ?g ?h ?m1 ?m2 0) (<> ?m1 ?m2))
((corr-val.- ?f ?g ?h ?m1 ?d1 0 ?d2 ?m3 ?d3) (<> ?m1 ?m3))
((corr-mag.- ?f ?g ?h ?m1 0 ?m3) (<> ?m1 ?m3))

((corr-val.- ?f ?g ?h ?m1 ?d1 ?m2 ?d2 ?m3 ?d3) (> ?m2 ?m1) (pos ?m3))
((corr-mag.- ?f ?g ?h ?m1 ?m2 ?m3) (> ?m2 ?m1) (pos ?m3))
((corr-val.- ?f ?g ?h ?m1 ?d1 ?m2 ?d2 ?m3 ?d3) (> ?m1 ?m2) (neg ?m3))
((corr-mag.- ?f ?g ?h ?m1 ?m2 ?m3) (> ?m1 ?m2) (neg ?m3))

((corr-val.- ?f ?g ?h 0 ?d1 ?m2 ?d2 ?m3 ?d3) (pos ?m2) (pos ?m3))
((corr-mag.- ?f ?g ?h 0 ?m2 ?m3) (pos ?m2) (pos ?m3))
((corr-val.- ?f ?g ?h 0 ?d1 ?m2 ?d2 ?m3 ?d3) (neg ?m2) (neg ?m3))
((corr-mag.- ?f ?g ?h 0 ?m2 ?m3) (neg ?m2) (neg ?m3))
))

(setf *assumption-nogoods* nil)
```

C.2.2 Temperature Controller

Knowledge Base The following axioms are specific to the temperature controller:

```
(setf *brules* '(
((qval ti ?m1 ?d1 ?t) <- (norm s) (qval ti-ob ?m1 ?d1 ?t))
((qval ts ?m1 ?d1 ?t) <- (norm k) (qval ts-ob ?m1 ?d1 ?t))
((qval e ?m3 ?d3 ?t) <- (norm c1) (qval ts ?m1 ?d1 ?t) (qval ti ?m2 ?d2 ?t)
                        (holds.- ts ti e ?m1 ?d1 ?m2 ?d2 ?m3 ?d3))
((qval a ?m2 ?d1 ?t) <- (norm c2) (qval e ?m1 ?d1 ?t)
                        (holds.m0+ e a ?m1 ?d1 ?m2 ?d1))
((qval p ?m3 ?d3 ?t) <- (norm o) (qval p-ob ?m1 ?d1 ?t) (qval w ?m2 ?d2 ?t)
                        (holds.* p-ob w p ?m1 ?d1 ?m2 ?d2 ?m3 ?d3))
((qval hfin ?m3 ?d3 ?t) <- (norm e) (qval a ?m1 ?d1 ?t) (qval p ?m2 ?d2 ?t)
                           (holds.* a p hfin ?m1 ?d1 ?m2 ?d2 ?m3 ?d3))

((qval ti 0-std ?t) <- (stuck-at-0-std s) (qval ti-ob ?m1 ?d1 ?t))
((qval ts 0 std ?t) <- (stuck-at-0-std k) (qval ts-ob ?m1 ?d1 ?t))
((qval e 0 std ?t) <- (stuck-at-0-std c1) (qval ts ?m1 ?d1 ?t)
                        (qval ti ?m2 ?d2 ?t))
((qval a 0 std ?t) <- (stuck-at-0-std c2) (qval e ?m1 ?d1 ?t))
((qval p 0 std ?t) <- (stuck-at-0-std o) (qval p-ob ?m1 ?d1 ?t)
                        (qval w ?m2 ?d2 ?t))
((qval hfin 0 std ?t) <- (stuck-at-0-std e) (qval a ?m1 ?d1 ?t)
                           (qval p ?m2 ?d2 ?t))

((qval ti roomtemp std ?t) <- (stuck-at-roomtemp-std s)
                                (qval ti-ob ?m1 ?d1 ?t) (pos roomtemp))
((qval e ?m1 ?d1 ?t) <- (stuck-at-1st-in c1)
                          (qval ts ?m1 ?d1 ?t) (qval ti ?m2 ?d2 ?t))
((qval e ?m2 ?d2 ?t) <- (stuck-at-2nd-in c1)
                          (qval ts ?m1 ?d1 ?t) (qval ti ?m2 ?d2 ?t))
((qval p ?m1 ?d1 ?t) <- (stuck-at-1st-in o)
                          (qval p-ob ?m1 ?d1 ?t) (qval w ?m2 ?d2 ?t))

((qval ti-ob ?m1 ?d1 ?t) <- (given-qval ti-ob ?m1 ?d1 ?t))
((qval ts-ob ?m1 ?d1 ?t) <- (given-qval ts-ob ?m1 ?d1 ?t))
((qval p-ob ?m1 ?d1 ?t) <- (given-qval p-ob ?m1 ?d1 ?t))
((qval w ?m1 ?d1 ?t) <- (given-qval w ?m1 ?d1 ?t))
))

(setf *corr-mags* nil)

(setf *inter-batch-beam-width* 40)
(setf *intra-batch-beam-width* 40)
(setf *bchain-depth* 6)
(setf *caching* t)
(setf *factoring* nil)
(setf *remove-superset?* nil)
(setf *explanation-eval-metric* #'diag-simplicity)
(setf *compute-estimate-fn* #'diag-compute-estimate)
(setf *combine-estimates-fn* #'diag-combine-estimates)

(setf *predicate-specific-abduction* t)
(setf *assumable-predicates*
      '(norm stuck-at-0-std stuck-at-1st-in stuck-at-2nd-in
        stuck-at-roomtemp-std given-qval
        pos neg corr-mag.m0+
        corr-mag.- corr-val.- corr-mag.* corr-val.*
        corr-mag./ corr-val./ corr-val.d/dt <> >))
(setf *free-assumption-predicates*
      '(norm given-qval
        pos neg corr-mag.m0+
        corr-mag.- corr-val.- corr-mag.* corr-val.*
        corr-mag./ corr-val./ corr-val.d/dt <> >))
(setf *fault-mode-predicates*
      '(stuck-at-0-std stuck-at-1st-in
        stuck-at-2nd-in stuck-at-roomtemp-std))
(setf *components* '(s k c1 c2 o e))
```

Test Data For each of the 10 scenarios tested, we give: (1) the faults present; and (2) the input atoms representing the dynamic behavior.

```
#1: ((stuck-at-0-std k) (stuck-at-0-std o))

((pos roomtemp) (qval ts-ob roomtemp std t1) (qval ti-ob roomtemp dec t1)
 (pos psup) (qval p-ob psup std t1) (pos on) (qval w on std t1)
 (qval hfin 0 std t1) (qval ts-ob roomtemp std t2) (pos cold)
 (> roomtemp cold) (qval ti-ob cold std t2) (qval p-ob psup std t2)
 (qval w on std t2) (qval hfin 0 std t2) (qval ts 0 std t2)
 (qval ti cold std t2) (neg e-1) (qval e e-1 std t2)
 (neg a-1) (qval a a-1 std t2) (qval p 0 std t2))

#2: ((stuck-at-0-std s) (stuck-at-0-std e))

((pos roomtemp) (qval ts-ob roomtemp std t1) (qval ti-ob roomtemp dec t1)
 (pos psup) (qval p-ob psup std t1) (pos on) (qval w on std t1)
 (qval hfin 0 std t1) (qval ts-ob roomtemp std t2) (pos cold)
 (> roomtemp cold) (qval ti-ob cold std t2) (qval p-ob psup std t2)
 (qval w on std t2) (qval hfin 0 std t2) (qval ts roomtemp std t2)
 (qval ti 0 std t2) (qval e roomtemp std t2) (pos a-0)
 (qval a a-0 std t2) (qval p psup std t2))

#3: ((stuck-at-1st-in c1) (stuck-at-0-std e))

((pos roomtemp) (qval ts-ob roomtemp std t1) (qval ti-ob roomtemp dec t1)
 (pos psup) (qval p-ob psup std t1) (pos on) (qval w on std t1)
 (qval hfin 0 std t1) (qval ts-ob roomtemp std t2) (pos cold)
 (> roomtemp cold) (qval ti-ob cold std t2) (qval p-ob psup std t2)
 (qval w on std t2) (qval hfin 0 std t2) (qval ts roomtemp std t2)
 (qval ti cold std t2) (qval e roomtemp std t2) (pos a-0)
 (qval a a-0 std t2) (qval p psup std t2))

#4: ((stuck-at-0-std s) (stuck-at-0-std c2))

((pos roomtemp) (qval ts-ob roomtemp std t1) (qval ti-ob roomtemp dec t1)
 (pos psup) (qval p-ob psup std t1) (pos on) (qval w on std t1)
 (qval hfin 0 std t1) (qval ts-ob roomtemp std t2) (pos cold)
 (> roomtemp cold) (qval ti-ob cold std t2) (qval p-ob psup std t2)
 (qval w on std t2) (qval hfin 0 std t2) (qval ts roomtemp std t2)
 (qval ti 0 std t2) (qval e roomtemp std t2) (qval a 0 std t2)
 (qval p psup std t2))

#5: ((stuck-at-0-std c1) (stuck-at-0-std c2))

((pos roomtemp) (qval ts-ob roomtemp std t1) (qval ti-ob roomtemp dec t1)
 (pos psup) (qval p-ob psup std t1) (pos on) (qval w on std t1)
 (qval hfin 0 std t1) (qval ts-ob roomtemp std t2) (pos cold)
 (> roomtemp cold) (qval ti-ob cold std t2) (qval p-ob psup std t2)
 (qval w on std t2) (qval hfin 0 std t2) (qval ts roomtemp std t2)
 (qval ti cold std t2) (qval e 0 std t2) (qval a 0 std t2)
 (qval p psup std t2))

#6: ((stuck-at-0-std c1) (stuck-at-0-std o))

((pos roomtemp) (qval ts-ob roomtemp std t1) (qval ti-ob roomtemp dec t1)
 (pos psup) (qval p-ob psup std t1) (pos on) (qval w on std t1)
 (qval hfin 0 std t1) (qval ts-ob roomtemp std t2) (pos cold)
 (> roomtemp cold) (qval ti-ob cold std t2) (qval p-ob psup std t2)
 (qval w on std t2) (qval hfin 0 std t2) (qval ts roomtemp std t2)
 (qval ti cold std t2) (qval e 0 std t2) (qval a 0 std t2)
 (qval p 0 std t2))

#7: ((stuck-at-1st-in c1) (stuck-at-0-std o))

((pos roomtemp) (qval ts-ob roomtemp std t1) (qval ti-ob roomtemp dec t1)
 (pos psup) (qval p-ob psup std t1) (pos on) (qval w on std t1)
 (qval hfin 0 std t1) (qval ts-ob roomtemp std t2) (pos cold)
 (> roomtemp cold) (qval ti-ob cold std t2) (qval p-ob psup std t2)
 (qval w on std t2) (qval hfin 0 std t2) (qval ts roomtemp std t2)
 (qval ti cold std t2) (qval e roomtemp std t2) (pos a-0)
 (qval a a-0 std t2) (qval p 0 std t2))

#8: ((stuck-at-roomtemp-std s))

((pos roomtemp) (qval ts-ob roomtemp std t1) (qval ti-ob roomtemp dec t1)
 (pos psup) (qval p-ob psup std t1) (pos on) (qval w on std t1)
 (qval hfin 0 std t1) (qval ts-ob roomtemp std t2) (pos cold)
 (> roomtemp cold) (qval ti-ob cold std t2) (qval p-ob psup std t2)
 (qval w on std t2) (qval hfin 0 std t2) (qval ts roomtemp std t2)
 (qval ti roomtemp std t2) (qval e 0 std t2) (qval a 0 std t2)
 (qval p psup std t2))

#9: ((stuck-at-0-std o))

((pos roomtemp) (qval ts-ob roomtemp std t1) (qval ti-ob roomtemp dec t1)
 (pos psup) (qval p-ob psup std t1) (pos on) (qval w on std t1)
 (qval hfin 0 std t1) (qval ts-ob roomtemp std t2) (pos cold)
 (> roomtemp cold) (qval ti-ob cold std t2) (qval p-ob psup std t2)
 (qval w on std t2) (qval hfin 0 std t2) (qval ts roomtemp std t2)
 (qval ti cold std t2) (pos e-0) (qval e e-0 std t2) (pos a-0)
 (qval a a-0 std t2) (qval p 0 std t2))

#10: ((stuck-at-0-std c2))

((pos roomtemp) (qval ts-ob roomtemp std t1) (qval ti-ob roomtemp dec t1)
 (pos psup) (qval p-ob psup std t1) (pos on) (qval w on std t1)
 (qval hfin 0 std t1) (qval ts-ob roomtemp std t2) (pos cold)
 (> roomtemp cold) (qval ti-ob cold std t2) (qval p-ob psup std t2)
 (qval w on std t2) (qval hfin 0 std t2) (qval ts roomtemp std t2)
 (qval ti cold std t2) (pos e-0) (qval e e-0 std t2) (qval a 0 std t2)
 (qval p psup std t2))
```

C.2.3 The Water Balance System of the Human Kidney

Knowledge Base The following axioms are specific to the kidney water balance system:

```
(setf *brules* '(
((qval cna ?m3 ?d3 ?t) <- (norm c1) (qval anp ?m1 ?d1 ?t)
                           (qval awp ?m2 ?d2 ?t)
                           (holds./ anp awp cna ?m1 ?d1 ?m2 ?d2 ?m3 ?d3))
((qval cnh ?m2 ?d1 ?t) <- (norm c2) (qval awp ?m1 ?d1 ?t)
                           (holds.m0+ awp cnh ?m1 ?d1 ?m2 ?d1))
((qval fpu ?m2 ?d1 ?t) <- (norm c3) (qval cnh ?m1 ?d1 ?t)
                           (holds.m0+ cnh fpu ?m1 ?d1 ?m2 ?d1))
((qval cadh ?m2 ?d1 ?t) <- (norm c4) (qval cna ?m1 ?d1 ?t)
                            (holds.m0+ cna cadh ?m1 ?d1 ?m2 ?d1))
((qval rfup ?m2 ?d1 ?t) <- (norm c5) (qval cadh ?m1 ?d1 ?t)
                            (holds.m0+ cadh rfup ?m1 ?d1 ?m2 ?d1))
((qval nfpu ?m3 ?d3 ?t) <- (norm c6) (qval fpu ?m1 ?d1 ?t)
                            (qval rfup ?m2 ?d2 ?t)
                            (holds.- fpu rfup nfpu ?m1 ?d1 ?m2 ?d2 ?m3 ?d3))
((qval nfop ?m3 ?d3 ?t) <- (norm c7) (qval nfip ?m1 ?d1 ?t)
                            (qval nfpu ?m2 ?d2 ?t)
                            (holds.- nfip nfpu nfop ?m1 ?d1 ?m2 ?d2 ?m3 ?d3))

((qval cnh cnh- std ?t) <- (stuck-at-low-std c2) (qval awp ?m1 ?d1 ?t)
                            (pos cnh-))
((qval cnh cnh+ std ?t) <- (stuck-at-high-std c2) (qval awp ?m1 ?d1 ?t)
                            (pos cnh+))

((qval fpu fpu- std ?t) <- (stuck-at-low-std c3) (qval cnh ?m1 ?d1 ?t)
                            (pos fpu-))
((qval fpu fpu+ std ?t) <- (stuck-at-high-std c3) (qval cnh ?m1 ?d1 ?t)
                            (pos fpu+))

((qval cadh cadh- std ?t) <- (stuck-at-low-std c4) (qval cna ?m1 ?d1 ?t)
                              (pos cadh-))
((qval cadh cadh+ std ?t) <- (stuck-at-high-std c4) (qval cna ?m1 ?d1 ?t)
                              (pos cadh+))

((qval rfup rfup- std ?t) <- (stuck-at-low-std c5) (qval cadh ?m1 ?d1 ?t)
                              (pos rfup-))
((qval rfup rfup+ std ?t) <- (stuck-at-high-std c5) (qval cadh ?m1 ?d1 ?t)
                              (pos rfup+))

((qval anp ?m1 ?d1 ?t) <- (given-qval anp ?m1 ?d1 ?t))
((qval awp ?m1 ?d1 ?t) <- (given-qval awp ?m1 ?d1 ?t))
((qval nfip ?m1 ?d1 ?t) <- (given-qval nfip ?m1 ?d1 ?t))))

(setf *corr-mags*
      '((corr-mag.m0+ awp cnh awp* cnh*)
        (corr-mag.m0+ awp cnh awp+ cnh+)
        (corr-mag.m0+ awp cnh awp- cnh-)
        (corr-mag.m0+ cnh fpu cnh* fpu*)
        (corr-mag.m0+ cnh fpu cnh+ fpu+)
        (corr-mag.m0+ cnh fpu cnh- fpu-)
        (corr-mag.m0+ cna cadh cna* cadh*)
        (corr-mag.m0+ cna cadh cna+ cadh+)
        (corr-mag.m0+ cna cadh cna- cadh-)
        (corr-mag.m0+ cadh rfup cadh* rfup*)
        (corr-mag.m0+ cadh rfup cadh+ rfup+)
        (corr-mag.m0+ cadh rfup cadh- rfup-)))
```

```
(setf *inter-batch-beam-width* 30)
(setf *intra-batch-beam-width* 30)
(setf *bchain-depth* 7)
(setf *caching* t)
(setf *factoring* nil)
(setf *remove-superset?* nil)
(setf *explanation-eval-metric* #'diag-simplicity)
(setf *compute-estimate-fn* #'diag-compute-estimate)
(setf *combine-estimates-fn* #'diag-combine-estimates)

(setf *predicate-specific-abduction* t)
(setf *assumable-predicates*
      '(norm stuck-at-high-std stuck-at-low-std given-qval
        pos neg corr-mag.m0+
        corr-mag.- corr-val.- corr-mag.* corr-val.*
        corr-mag./ corr-val./ corr-val.d/dt <> >))
(setf *free-assumption-predicates*
      '(norm given-qval
        pos neg corr-mag.m0+
        corr-mag.- corr-val.- corr-mag.* corr-val.*
        corr-mag./ corr-val./ corr-val.d/dt <> >))
(setf *fault-mode-predicates*
      '(stuck-at-high-std stuck-at-low-std))
(setf *components* '(c1 c2 c3 c4 c5 c6 c7))
```

Test Data

```
#1: ((stuck-at-high-std c4))

((pos anp*) (qval anp anp* std t1) (pos awp+) (qval awp awp+ inc t1)
 (pos nfip+) (qval nfip nfip+ std t1) (pos n-0) (qval nfop n-0 dec t1)
 (qval anp anp* std t2) (pos a-0) (qval awp a-0 std t2)
 (qval nfip nfip+ std t2) (qval nfop 0 std t2) (pos c-0)
 (qval cna c-0 std t2) (pos c-1) (qval cnh c-1 std t2) (pos cadh+)
 (qval cadh cadh+ std t2) (pos f-0) (qval fpu f-0 std t2) (pos rfup+)
 (qval rfup rfup+ std t2))

#2: ((stuck-at-low-std c4))

((pos anp*) (qval anp anp* std t1) (pos awp-) (qval awp awp- inc t1)
 (pos nfip+) (qval nfip nfip+ std t1) (pos n-0) (qval nfop n-0 dec t1)
 (qval anp anp* std t2) (pos awp*) (qval awp awp* inc t2)
 (qval nfip nfip+ std t2) (pos n-1) (qval nfop n-1 dec t2) (pos cna*)
 (qval cna cna* dec t2) (pos cnh*) (qval cnh cnh* inc t2) (pos cadh-)
 (qval cadh cadh- std t2) (pos fpu*) (qval fpu fpu* inc t2)
 (pos rfup-) (qval rfup rfup- std t2))

#3: ((stuck-at-high-std c5))

((pos anp*) (qval anp anp* std t1) (pos awp+) (qval awp awp+ inc t1)
 (pos nfip+) (qval nfip nfip+ std t1) (pos n-0) (qval nfop n-0 dec t1)
 (qval anp anp* std t2) (pos a-0) (qval awp a-0 std t2)
 (qval nfip nfip+ std t2) (qval nfop 0 std t2) (pos c-0)
 (qval cna c-0 std t2) (pos c-1) (qval cnh c-1 std t2) (pos c-2)
 (qval cadh c-2 std t2) (pos f-0) (qval fpu f-0 std t2) (pos rfup+)
 (qval rfup rfup+ std t2))

#4: ((stuck-at-low-std c5))

((pos anp*) (qval anp anp* std t1) (pos awp-) (qval awp awp- inc t1)
 (pos nfip+) (qval nfip nfip+ std t1) (pos n-0) (qval nfop n-0 dec t1)
 (qval anp anp* std t2) (pos awp*) (qval awp awp* inc t2)
 (qval nfip nfip+ std t2) (pos n-1) (qval nfop n-1 dec t2)
 (pos cna*) (qval cna cna* dec t2) (pos cnh*) (qval cnh cnh* inc t2)
 (pos cadh*) (qval cadh cadh* dec t2) (pos fpu*) (qval fpu fpu* inc t2)
 (pos rfup-) (qval rfup rfup- std t2))

#5: ((stuck-at-high-std c2))

((pos anp*) (qval anp anp* std t1) (pos awp-) (qval awp awp- inc t1)
 (pos nfip+) (qval nfip nfip+ std t1) (pos n-0) (qval nfop n-0 dec t1)
 (qval anp anp* std t2) (pos awp*) (qval awp awp* inc t2)
 (qval nfip nfip+ std t2) (pos n-1) (qval nfop n-1 dec t2)
 (pos cna*) (qval cna cna* dec t2) (pos cnh+) (qval cnh cnh+ std t2)
 (pos cadh*) (qval cadh cadh* dec t2) (pos fpu+) (qval fpu fpu+ std t2)
 (pos rfup*) (qval rfup rfup* dec t2))

#6: ((stuck-at-low-std c2))

((pos anp*) (qval anp anp* std t1) (pos awp+) (qval awp awp+ inc t1)
 (pos nfip+) (qval nfip nfip+ std t1) (pos n-0) (qval nfop n-0 dec t1)
 (qval anp anp* std t2) (pos a-0) (qval awp a-0 std t2)
 (qval nfip nfip+ std t2) (qval nfop 0 std t2) (pos c-0)
 (qval cna c-0 std t2) (pos cnh-) (qval cnh cnh- std t2) (pos c-1)
 (qval cadh c-1 std t2) (pos fpu-) (qval fpu fpu- std t2) (pos r-0)
 (qval rfup r-0 std t2))

#7: ((stuck-at-high-std c3))

((pos anp*) (qval anp anp* std t1) (pos awp-) (qval awp awp- inc t1)
 (pos nfip+) (qval nfip nfip+ std t1) (pos n-0) (qval nfop n-0 dec t1)
 (qval anp anp* std t2) (pos awp*) (qval awp awp* inc t2)
 (qval nfip nfip+ std t2) (pos n-1) (qval nfop n-1 dec t2)
 (pos cna*) (qval cna cna* dec t2) (pos cnh*) (qval cnh cnh* inc t2)
 (pos cadh*) (qval cadh cadh* dec t2) (pos fpu+) (qval fpu fpu+ std t2)
 (pos rfup*) (qval rfup rfup* dec t2))

#8: ((stuck-at-low-std c3))

((pos anp*) (qval anp anp* std t1) (pos awp+) (qval awp awp+ inc t1)
 (pos nfip+) (qval nfip nfip+ std t1) (pos n-0) (qval nfop n-0 dec t1)
 (qval anp anp* std t2) (pos a-0) (qval awp a-0 std t2)
 (qval nfip nfip+ std t2) (qval nfop 0 std t2) (pos c-0)
 (qval cna c-0 std t2) (pos c-1) (qval cnh c-1 std t2) (pos c-2)
 (qval cadh c-2 std t2) (pos fpu-) (qval fpu fpu- std t2)
 (pos r-0) (qval rfup r-0 std t2))
```

```
#9: ((stuck-at-high-std c3) (stuck-at-low-std c4))

((pos anp*) (qval anp anp* std t1) (pos awp*) (qval awp awp* std t1)
 (pos nfip+) (qval nfip nfip+ std t1) (qval nfop 0 std t1)
 (pos cna*) (qval cna cna* std t1) (pos cnh*) (qval cnh cnh* std t1)
 (pos cadh-) (qval cadh cadh- std t1) (pos fpu+) (qval fpu fpu+ std t1)
 (pos rfup-) (qval rfup rfup- std t1))

#10: ((stuck-at-high-std c3) (stuck-at-low-std c5))

((pos anp*) (qval anp anp* std t1) (pos awp*) (qval awp awp* std t1)
 (pos nfip+) (qval nfip nfip+ std t1) (qval nfop 0 std t1)
 (pos cna*) (qval cna cna* std t1) (pos cnh*) (qval cnh cnh* std t1)
 (pos cadh*) (qval cadh cadh* std t1) (pos fpu+) (qval fpu fpu+ std t1)
 (pos rfup-) (qval rfup rfup- std t1))
```

BIBLIOGRAPHY

[Aho *et al.*, 1974] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, MA, 1974.

[Allemang *et al.*, 1987] Dean Allemang, Michael C. Tanner, Tom Bylander, and John R. Josephson. Computational complexity of hypothesis assembly. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 1112–1117, Milan, Italy, 1987.

[Allen and Perrault, 1980] James F. Allen and C. Raymond Perrault. Analyzing intention in utterances. *Artificial Intelligence*, 15(3):143–178, 1980.

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

[Alterman, 1985] Richard Alterman. A dictionary based on concept coherence. *Artificial Intelligence*, 25(2):153–186, 1985.

[Bylander *et al.*, 1989] Tom Bylander, Dean Allemang, Michael C. Tanner, and John R. Josephson. Some results concerning the computational complexity of abduction. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 44–54, Toronto, Ontario, Canada, 1989.

[Charniak and Goldman, 1989] Eugene Charniak and Robert P. Goldman. A semantics for probabilistic quantifier-free first-order languages, with particular application to story understanding. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, 1989.

[Charniak and Goldman, 1991] Eugene Charniak and Robert Goldman. A probabilistic model of plan recognition. In *Proceedings of the National Conference on Artificial Intelligence*, pages 160–165, Anaheim, CA, 1991.

[Charniak and McDermott, 1985] Eugene Charniak and Drew McDermott. *Introduction to Artificial Intelligence.* Addison Wesley, Reading, MA, 1985.

[Charniak and Shimony, 1990] Eugene Charniak and Solomon E. Shimony. Probabilistic semantics for cost based abduction. In *Proceedings of the National Conference on Artificial Intelligence*, pages 106–111, Boston, MA, 1990.

[Charniak, 1978] Eugene Charniak. With a spoon in hand this must be the eating frame. In *Theoretical Issues in Natural Language Processing 2*, pages 187–193, Urbana, IL, 1978.

[Charniak, 1986] Eugene Charniak. A neat theory of marker passing. In *Proceedings of the National Conference on Artificial Intelligence*, pages 584–588, Philadelphia, PA, 1986.

[Charniak, 1987] Eugene Charniak. Logic and explanation. *Computational Intelligence*, 3:172–174, 1987.

[Charniak, 1988] Eugene Charniak. Motivation analysis, abductive unification, and nonmonotonic equality. *Artificial Intelligence*, 34:275–295, 1988.

[Cox and Pietrzykowski, 1986] P. T. Cox and T. Pietrzykowski. Causes for events: Their computation and applications. In *Proceedings of the Eighth International Conference on Automated Deduction*, pages 608–621. Springer-Verlag, 1986. Lecture Notes in Computer Science 230.

[Cox and Pietrzykowski, 1987] P. T. Cox and T. Pietrzykowski. General diagnosis by abductive inference. In *Proceedings of the 1987 Symposium on Logic Programming*, pages 183–189, 1987.

[Cullingford, 1978] Richard E. Cullingford. Script application: Computer understanding of newspaper stories. Technical Report 116, Department of Computer Science, Yale University, New Haven, CT, January 1978.

[Davis, 1984] Randall Davis. Diagnostic reasoning based on structure and behavior. *Artificial Intelligence*, 24:347–410, 1984.

[de Kleer and Williams, 1987] Johan de Kleer and Brian C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32:97–130, 1987.

[de Kleer and Williams, 1989] Johan de Kleer and Brian C. Williams. Diagnosis with behavioral modes. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1324–1330, Detroit, MI, 1989.

[de Kleer, 1986] Johan de Kleer. An assumption-based TMS. *Artificial Intelligence*, 28:127–162, 1986.

[Doyle, 1979] Jon Doyle. A truth maintenance system. *Artificial Intelligence*, 12:231–272, 1979.

[Dressler and Farquhar, 1990] Oskar Dressler and Adam Farquhar. Putting the problem solver back in the driver's seat: Contextual control of the ATMS. In *Proceedings of the Second Model-Based Reasoning Workshop*, Boston, MA, 1990.

[Dvorak and Kuipers, 1989] Daniel Dvorak and Benjamin J. Kuipers. Model-based monitoring of dynamic systems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1238–1243, Detroit, MI, 1989.

[Dvorak, 1992] Daniel Dvorak. *Monitoring and Diagnosis of Continuous Dynamic Systems Using Semiquantitative Simulation.* PhD thesis, Department of Computer Sciences, University of Texas at Austin, Austin, TX, May 1992.

[Eiselt, 1987] Kurt P. Eiselt. Recovering from erroneous inferences. In *Proceedings of the National Conference on Artificial Intelligence*, pages 540–544, Seattle, Washington, 1987.

[Elkan, 1989] Charles Elkan. Conspiracy numbers and caching for searching and/or trees and theorem-proving. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 341–346, Detroit, MI, 1989.

[Forbus and de Kleer, 1988] Kenneth D. Forbus and Johan de Kleer. Focusing the ATMS. In *Proceedings of the National Conference on Artificial Intelligence*, pages 193–198, St. Paul, Minnesota, 1988.

[Forbus, 1984] Kenneth D. Forbus. Qualitative process theory. *Artificial Intelligence*, 24:85–168, 1984.

[Garey and Johnson, 1979] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, San Francisco, CA, 1979.

[Genesereth, 1984] Michael R. Genesereth. The use of design descriptions in automated diagnosis. *Artificial Intelligence*, 24:411–436, 1984.

[Ginsberg, 1989] Matthew L. Ginsberg. A circumscriptive theorem prover. *Artificial Intelligence*, 39:209–230, 1989.

[Goldman, 1990] Robert P. Goldman. *A Probabilistic Approach to Language Understanding.* PhD thesis, Department of Computer Science, Brown University, Providence, RI, December 1990. Technical Report CS-90-34.

[Granger, 1980a] Richard Horace Granger, Jr. *Adaptive Understanding: Correcting Erroneous Inferences.* PhD thesis, Department of Computer Science, Yale University, New Haven, CT, January 1980. Technical Report 171.

[Granger, 1980b] Richard Horace Granger, Jr. When expectation fails: Towards a self-correcting inference system. In *Proceedings of the First National Conference on Artificial Intelligence*, pages 301–305, Stanford, California, 1980.

[Grice, 1975] H. P. Grice. Logic and conversation. In P. Cole and J. Morgan, editors, *Syntax and Semantics 3: Speech Acts*, pages 41–58. Academic Press, New York, 1975.

[Hamscher, 1990] Walter Hamscher. XDE: Diagnosing devices with hierarchic structure and known component failure modes. In *Proceedings of the Sixth IEEE Conference on Artificial Intelligence Applications*, pages 48–54, Santa Barbara, CA, 1990.

[Hickman and Larkin, 1990] Angela Kennedy Hickman and Jill H. Larkin. Internal analogy: A model of transfer within problems. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, pages 53–60, Cambridge, MA, 1990.

[Hobbs *et al.*, 1988] Jerry R. Hobbs, Mark E. Stickel, Paul Martin, and Douglas Edwards. Interpretation as abduction. In *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics*, pages 95–103, Buffalo, New York, 1988.

[Kautz and Allen, 1986] Henry A. Kautz and James F. Allen. Generalized plan recognition. In *Proceedings of the National Conference on Artificial Intelligence*, pages 32–37, Philadelphia, PA, 1986.

[Kautz, 1987] Henry A. Kautz. *A Formal Theory of Plan Recognition.* PhD thesis, Department of Computer Science, University of Rochester, Rochester, NY, May 1987. Technical Report 215.

[Konolige, 1992] Kurt Konolige. Abduction versus closure in causal theories. *Artificial Intelligence*, 53:255–272, 1992.

[Kuipers and Berleant, 1988] Benjamin J. Kuipers and Daniel Berleant. Using incomplete quantitative knowledge in qualitative reasoning. In *Proceedings of the National Conference on Artificial Intelligence*, pages 324–329, Saint Paul, Minnesota, 1988.

[Kuipers, 1984] Benjamin J. Kuipers. Commonsense reasoning about causality: Deriving behavior from structure. *Artificial Intelligence*, 24:169–203, 1984.

[Kuipers, 1985] Benjamin J. Kuipers. Qualitative simulation in medical physiology: A progress report. Technical Report MIT/LCS/TM-280, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, June 1985.

[Kuipers, 1986] Benjamin J. Kuipers. Qualitative simulation. *Artificial Intelligence*, 29:289–338, 1986.

[Kuipers, 1991] Benjamin J. Kuipers. Qualitative reasoning: Modeling and simulation with incomplete knowledge. Book draft, May 1991.

[Laird *et al.*, 1984] John E. Laird, Paul S. Rosenbloom, and Allen Newell. Towards chunking as a general learning mechanism. In *Proceedings of the National Conference on Artificial Intelligence*, pages 188–192, Austin, TX, 1984.

[Lauritzen and Spiegelhalter, 1988] S. L. Lauritzen and David J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society B*, 50(2):157–224, 1988.

[Lehnert and Sundheim, 1991] Wendy Lehnert and Beth Sundheim. A performance evaluation of text-analysis technologies. *AI Magazine*, 12(3):81–94, 1991.

[Levesque, 1989] Hector J. Levesque. A knowledge-level account of abduction. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1061–1067, Detroit, MI, 1989.

[Litman and Allen, 1987] Diane J. Litman and James F. Allen. A plan recognition model for subdialogues in conversations. *Cognitive Science*, 11:163–200, 1987.

[McAllester, 1985] David McAllester. A widely used truth-maintenance system. Ai lab memo, MIT, Cambridge, MA, 1985.

[McCarthy, 1980] John McCarthy. Circumscription — a form of non-monotonic reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[McCune, 1990] William W. McCune. OTTER 2.0 users guide. Technical Report ANL-90/9, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois, March 1990.

[Minton, 1988] Steven Minton. Quantitative results concerning the utility of explanation-based learning. In *Proceedings of the National Conference on Artificial Intelligence*, pages 564–569, St. Paul, Minnesota, 1988.

[Murray, 1988] Kenneth S. Murray. KI: An experiment in automating knowledge integration. Technical Report AI88-90, Artificial Intelligence Laboratory, Department of Computer Sciences, The University of Texas at Austin, 1988.

[Ng and Mooney, 1989] Hwee Tou Ng and Raymond J. Mooney. Abductive explanation in text understanding: Some problems and solutions. Technical Report AI89-116, Artificial Intelligence Laboratory, Department of Computer Sciences, The University of Texas at Austin, October 1989.

[Ng and Mooney, 1990] Hwee Tou Ng and Raymond J. Mooney. On the role of coherence in abductive explanation. In *Proceedings of the National Conference on Artificial Intelligence*, pages 337–342, Boston, MA, 1990.

[Ng and Mooney, 1991a] Hwee Tou Ng and Raymond J. Mooney. An efficient first-order abduction system based on the ATMS. Technical Report AI91-151, Artificial Intelligence Laboratory, Department of Computer Sciences, The University of Texas at Austin, January 1991.

[Ng and Mooney, 1991b] Hwee Tou Ng and Raymond J. Mooney. An efficient first-order Horn-clause abduction system based on the ATMS. In *Proceedings of the National Conference on Artificial Intelligence*, pages 494–499, Anaheim, CA, 1991.

[Ng, 1990] Hwee Tou Ng. Model-based, multiple fault diagnosis of time-varying, continuous physical devices. In *Proceedings of the Sixth IEEE Conference on Artificial Intelligence Applications*, pages 9–15, Santa Barbara, CA, 1990. To appear in *Readings in Model-based Diagnosis*, edited by Walter Hamscher, Luca Console, and Johan de Kleer.

[Ng, 1991] Hwee Tou Ng. Model-based, multiple-fault diagnosis of dynamic, continuous physical devices. *IEEE Expert*, 6(6):38–43, 1991.

[Norvig and Wilensky, 1990] Peter Norvig and Robert Wilensky. A critical evaluation of commensurable abduction models for semantic interpretation. In *Proceedings of the Thirteenth International Conference on Computational Linguistics*, Helsinki, Finland, August 1990.

[Norvig, 1987] Peter Norvig. *Unified Theory of Inference for Text Understanding*. PhD thesis, Computer Science Division, University of California at Berkeley, Berkeley, CA, 1987. Technical Report No. UCB/CSD 87/339.

[Norvig, 1992] Peter Norvig. *Paradigms of Artificial Intelligence Programming: Case Studies in Common Lisp*. Morgan Kaufmann, San Mateo, CA, 1992.

[O'Rorke *et al.*, 1989] Paul O'Rorke, Steven Morris, and David Schulenburg. Theory formation by abduction: Initial results of a case study based on the chemical revolution. In *Proceedings of the Sixth International Workshop on Machine Learning*, 1989.

[Pearl, 1988] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.

[Peirce, 1958] Charles Sanders Peirce. *Collected Papers of Charles Sanders Peirce (1839–1914)*. Harvard University Press, Cambridge, MA, 1958.

[Peng and Reggia, 1990] Yun Peng and James A. Reggia. *Abductive Inference Models for Diagnostic Problem-Solving.* Springer-Verlag, New York, 1990.

[Pollack, 1989] Martha E. Pollack. Plan recognition beyond STRIPS. In *Working Notes of the Second AAAI Workshop on Plan Recognition*, Detroit, Michigan, August 1989.

[Poole, 1988] David Poole. Representing knowledge for logic-based diagnosis. In *Proceedings of the International Conference on Fifth Generation Computing Systems*, pages 1282–1290, Tokyo, Japan, 1988.

[Poole, 1989a] David Poole. A methodology for using a default and abductive reasoning system. Technical Report 89-20, Department of Computer Science, University of British Columbia, Vancouver, B.C., Canada, September 1989.

[Poole, 1989b] David Poole. Normality and faults in logic-based diagnosis. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1304–1310, Detroit, MI, 1989.

[Pople, 1973] Harry E. Pople, Jr. On the mechanization of abductive logic. In *Proceedings of the Third International Joint Conference on Artificial Intelligence*, pages 147–152, 1973.

[Reggia *et al.*, 1983] James A. Reggia, Dana S. Nau, and Pearl Y. Wang. Diagnostic expert systems based on a set covering model. *International Journal of Man-Machine Studies*, 19:437–460, 1983.

[Reggia *et al.*, 1985] James A. Reggia, Dana S. Nau, and Pearl Y. Wang. A formal model of diagnostic inference. I. problem formulation and decomposition. *Information Sciences*, 37:227–256, 1985.

[Reiter, 1987] Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[Schank and Abelson, 1977] Roger C. Schank and R. P. Abelson. *Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures.* Lawrence Erlbaum and Associates, Hillsdale, NJ, 1977.

[Selman and Levesque, 1990] Bart Selman and Hector J. Levesque. Abductive and default reasoning: A computational core. In *Proceedings of the National Conference on Artificial Intelligence*, pages 343–348, Boston, MA, 1990.

[Stickel, 1988a] Mark E. Stickel. A Prolog-like inference system for computing minimum-cost abductive explanations in natural-language interpretation. Technical Note 451, SRI International, September 1988.

[Stickel, 1988b] Mark E. Stickel. A Prolog technology theorem prover: Implementation by an extended Prolog compiler. *Journal of Automated Reasoning*, 4:353–380, 1988.

[Stickel, 1991] Mark E. Stickel. Upside-down meta-interpretation of the model elimination theorem-proving procedure for deduction and abduction. Technical Report TR-664, Institute for New Generation Computer Technology, Tokyo, Japan, May 1991.

[Struss and Dressler, 1989] Peter Struss and Oskar Dressler. Physical negation — integrating fault models into the general diagnostic engine. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1318–1323, Detroit, MI, 1989.

[Subramanian and Mooney, 1991] Siddarth Subramanian and Raymond J. Mooney. Combining abduction and theory revision. In *Proceedings of the First International Workshop on Multistrategy Learning*, pages 207–214, Harpers Ferry, W. Va., 1991.

[Subramanian, 1992] Siddarth Subramanian. Belief revision in the context of abductive explanation. dissertation proposal, 1992.

[Thagard, 1989] Paul Thagard. Explanatory coherence. *Behavioral and Brain Sciences*, 12(3):435–467, September 1989.

[Tuhrim *et al.*, 1991] Stanley Tuhrim, James Reggia, and Sharon Goodall. An experimental study of criteria for hypothesis plausibility. *Journal of Experimental and Theoretical Artificial Intelligence*, 3:129–144, 1991.

[Wilensky, 1978] Robert W. Wilensky. *Understanding Goal-Based Stories.* PhD thesis, Department of Computer Science, Yale University, New Haven, CT, September 1978. Technical Report 140.

[Wilensky, 1983] Robert W. Wilensky. *Planning and Understanding: A Computational Approach to Human Reasoning.* Addison-Wesley, Reading, MA, 1983.

VITA

Hwee Tou Ng was born in Singapore on November 27, 1963, the son of Choon Teck Ng and Swee Kiang Sim. In 1981, he completed his pre-university education at the National Junior College in Singapore, and received the Prime Minister's Book Prize, both in 1976 and 1982, for outstanding academic achievement and language proficiency. Subsequently, he was awarded a National Computer Board Undergraduate Scholarship to pursue an undergraduate degree in Computer Science oversees. He received his B.Sc. degree in Computer Science for Data Management (with high distinction) from the University of Toronto, Canada in August, 1985, and his M.S. degree in Computer Science from Stanford University in March 1987. He enrolled in the PhD program of the Department of Computer Sciences at the University of Texas at Austin in September 1987. In March 1990, he received the Best Student Paper Award for his paper "Model-Based, Multiple Fault Diagnosis of Time-Varying, Continuous Physical Devices" at the Sixth IEEE Conference on AI Applications. He married Boon Kheng Ang in May, 1990.

Permanent address: Block 16, Outram Hill
#02-40 Singapore 0316
Republic of Singapore

This dissertation was typeset[1] with LaTeX by the author.

[1] LaTeX document preparation system was developed by Leslie Lamport as a special version of Donald Knuth's TeX program for computer typesetting. TeX is a trademark of the American Mathematical Society. The LaTeX macro package for The University of Texas at Austin dissertation format was written by Khe-Sing The.