Default Reasoning and Diagnosis as Theory Formation

David L. Poole

Department of Computer Science
University of Waterloo
Waterloo, Ontario
Canada, N2L 3G1

Technical Report CS-86-08
March 1986

Default Reasoning and Diagnosis as Theory Formation

David L Poole

Logic Programming and Artificial Intelligence Group
Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1

ABSTRACT

We present a simple model theoretic semantics for both defaults and diagnosis. The semantics is based on normal first order model theory; instead of changing the logic, we propose to change the way in which the logic is used. Rather than deriving the consequences of our knowledge (finding what logically follows), we build falsifiable "scientific" theories which explain some set of observations. By using a predefined set of possible hypotheses which can be defaults or possible malfunctions or diseases, this idea subsumes the intuition behind Reiter's default logic and characterises model-based diagnosis. A prototype implementation, called *Theorist*, executes all of the examples given.

1. Introduction

There has been a perceived problem with using traditional logic for commonsense reasoning, as it is monotonic. There have been many proposals for augmenting classical logic to allow it to do non-monotonic reasoning (Reiter[80], McDermott and Doyle[80], McCarthy[80,84], Moore[83]). We argue that, rather than being a problem with logic, it is a problem of how we use logic.

Essentially we propose that, instead of viewing reasoning as deduction from our knowledge, it is better to view it as "scientific" theory formation (Popper[59], Quine and Ullian[78]), based on a set of user-given possible hypotheses. We show that Reiter's defaults and the notion of diagnosis from first principles can be explained in this framework. By allowing defaults and diagnosis within the same framework, we can build diagnostic systems which are based on a simple semantics, and can handle generalised and possibly erroneous or noisy knowledge.

We first give a syntax and a semantics for our logic, then some examples, and argue that the logic fits with our intuition behind defaults and diagnosis. Finally we describe an implementation and compare it to other diagnostic systems.

2. Syntax

Here we define the syntax of our default logic as an extension of the syntax of the first order predicate calculus.

The **symbols** in the language are the variables (in upper case), constants, function symbols, predicate symbols (lower case), standard logical connectives $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction), $\supset$ (implication), $\leftarrow$ (if), (,) (parentheses).

We first give standard definitions of predicate calculus syntax. A **term** is either a variable, constant symbol or a function symbol followed by a parenthesised list of terms. An **atomic symbol** (atom) is a predicate symbol followed by a list of terms. A **well formed formula** (wff) is an atomic symbol or $\neg w_1$, $(w_1 \wedge w_2)$, $(w_1 \vee w_2)$, $(w_1 \supset w_2)$, $(w_1 \leftarrow w_2)$, $(\forall v\ w_1)$, $(\exists v\ w_1)$ where w_1 and w_2 are well formed formulae, and v is a variable. Parentheses may be omitted when no ambiguity occurs.

Facts are defined by an expression of the form:

fact w;

where w is a wff. Free variables in w are assumed to be universally quantified. (If w is of the form of a clause "$L_0 \leftarrow L_1 \wedge \cdots \wedge L_n$" then we use the key word "*rule*" instead of "*fact*"; in the implementation this provides heuristic information.)

A **possible hypothesis** is an expression of the form

assume n: w;

where n is a term (the "name" of the possible hypothesis) and w is a wff. A default is said to be **closed** if there are no free variables in n.

For example the defaults "birds fly" is written as the open default

assume $birdsfly(X)$: $flies(X) \leftarrow bird(X)$;

An **instance** of default *assume* n: w is the wff formed by substituting in w ground terms for the free variables in n. The corresponding instance of n is the name of the instance. All other variables (in w) are assumed to be universally quantified.

2.1. Semantics

The intuitive idea is, given a set of observations to be explained, a set of facts known to be true, and a pool of possible hypotheses, to find a *theory* (a set of instances of possible hypotheses) which can be used to explain the observations (i.e., together with the fact implies the observations) and is consistent with the facts (i.e., does not predict anything known to be false). This should be viewed as a "scientific theory" based on a restricted set of possible hypotheses (see figure 1).

The model theoretic definition of the first order predicate logic says that for a set of closed wffs A and a closed wff w, $A \models w$ means that w is true in every model of A. That is, there is no interpretation in which every element of A is true and w is false, given the normal definition of the logical connectives.

Definition: given a set of wffs, F, (called the facts), and a set of defaults Δ, we say closed wff w is **explainable** if there is a finite set D of ground instances of elements of Δ such that

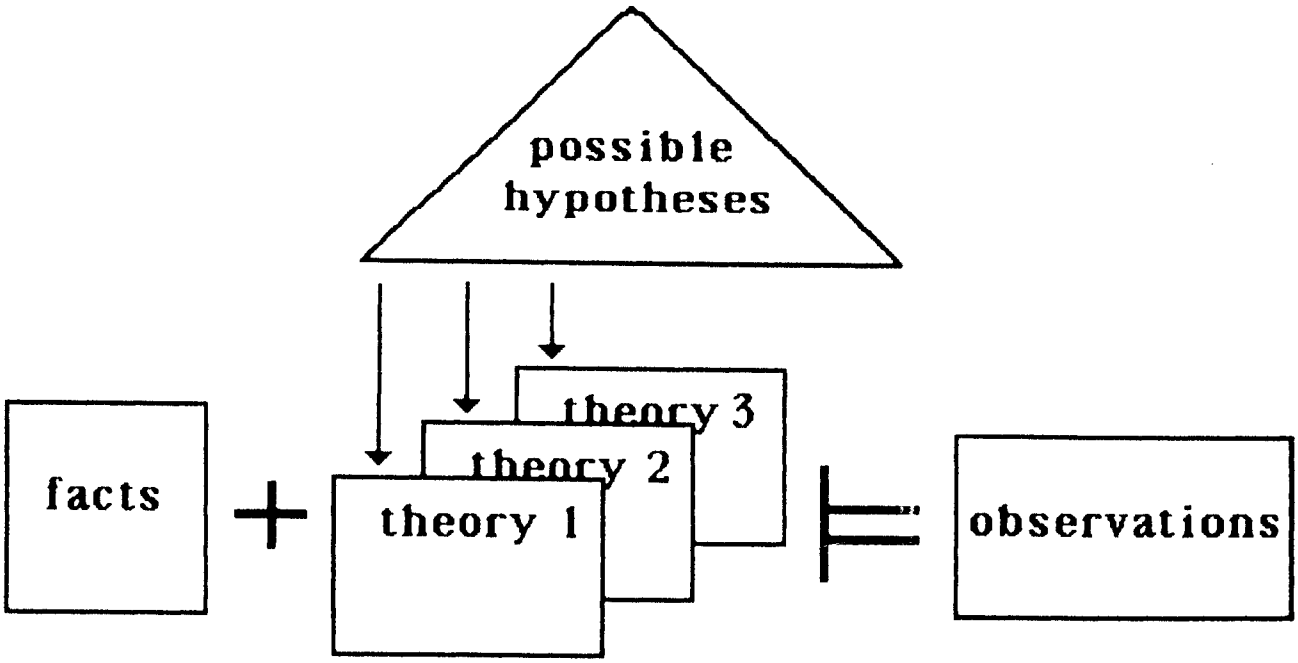

Figure 1 Architecture of Theorist

$F \cup D \models w$ and
$F \cup D$ is consistent

D is said to be the **theory that explains** w. D is a "scientific" theory, which is part of the "logical" theory which is the deductive closure of $F \cup D$.

3. Default Reasoning

In this section we show that if the possible hypotheses are defaults, the semantics above gives an account for default reasoning.

3.1. Example 1 - "birds fly"

Consider the statement "birds fly". This can be expressed as the default:

assume $birdsfly(X)$: $flies(X) \leftarrow bird(X)$;

This means that, for a particular value b of X, if $bird(b)$ can be proven then so can $flies(b)$, as long as the default has not been contradicted for that particular b. If $bird(b)$ and $\neg\, flies(b)$ are provable then the default is contradicted for that particular b.

Suppose that, together with the above assumption, we have the facts that emus are birds which do not fly, Polly is an emu and Tweety is a bird. This can be expressed as:

fact $emu(X) \supset bird(X)$;
fact $emu(X) \supset \neg\, flies(X)$;
fact $emu(polly)$;
fact $bird(tweety)$;

$flies(tweety)$ is explainable by the theory $\{birdsfly(tweety)\}$, which is consistent with the facts. $flies(polly)$ is potentially explainable by the theory $\{birdsfly(polly)\}$, but this is not consistent with the facts, as its negation can be proved. So $flies(polly)$ is not explainable.

3.2. A comparison with Reiter's logic

This section will make two comparisons with Reiter's default logic (Reiter[80]). One shows that the above logic is a restricted case of his normal defaults, and the second shows that the extra part of his defaults are not needed.

3.2.1. A restricted form of Reiter's normal defaults

The default schema

assume n: w;

(where the free variables in w are the same as those in n) is a syntactic variant of Reiter's normal default

$$\frac{:Mw}{w}$$

Reiter's logic for closed defaults is defined in terms of "extensions". In this section we explain extensions in terms of our logic.

First of all we define a partial order on theories; suppose D_1 and D_2 are theories, D_1 is less than D_2 (written $D_1 \subseteq D_2$) if D_1 is a subset of D_2. The following lemma trivially holds.

Lemma 1: (monotonicity) If $D_1 \subseteq D_2$ and $F \cup D_1 \models \alpha$ then $F \cup D_2 \models \alpha$

Define a **maximal theory** as a maximal element of $\{D{:}D$ *is a set of ground instances of* Δ *and* $F \cup D$ *is consistent*$\}$. Suppose $Th(A)$ is the set of all logical consequences of the set of axioms A. Define an **extension** to be $Th(D \cup F)$ where D is a maximal theory.

Note that a maximal theory is not necessarily a theory (as it may not be finite). It is countable (as the Herbrand universe is countable), but not necessarily finite. For example, assume a system with functions and the following rules about p:

assume $pn(X)$: $p(X)$
fact $\neg \forall X\ p(X)$

Its maximal theory includes the set of all conjunctions of elements of p such that not every value is in the conjunction.

Theorem 2: α is explainable $\equiv \alpha$ is in some extension.

Proof: If α is explainable then there exists D a set of instances of Δ such that $F \cup D$ is consistent, and $F \cup D \models \alpha$. By the definition of maximal, there exists a maximal theory E such that $D \subseteq E$. By the lemma 1, $F \cup E \models \alpha$, so $\alpha \in Th(F \cup E)$.

Conversely, suppose that α is in some extension. Then $\alpha \in Th(F \cup E)$ for some E. Let D be the (finite) subset of E used in a proof of α. Then α is explainable by the theory D.
Q.E.D.

The following theorem parallels the definition of a closed extension given in Reiter[80]. This shows that our definition of an extension in terms of maximal theories corresponds to his definition.

Theorem 3: Suppose E is an extension of $<F,\Delta>$. Then

(1) $F \subseteq E$

(2) $Th(E)=E$

(3) If *assume* n: α is a ground instance of a possible hypothesis, and $\neg \alpha \notin E$ then $\alpha \in E$

Furthermore E is minimal with respect to the above three properties.

Proof (1) follows since $F \subseteq Th(F \cup D)$ for any D. (2) follows from the definition of Th, since $Th(Th(S))=Th(S)$ for any S. (3) if "*assume* n: α" is a ground instance of a possible hypothesis, and $\neg \alpha \notin E$, then suppose $\alpha \notin E$. Then $E \cup \{\alpha\}$ is larger than E and $E \cup \{\alpha\}$ is consistent, as $\neg \alpha \notin Th(E)$, which contradicts the maximality of E.

To show E is minimal with respect to the above three properties, suppose $E' \subseteq E$; $E' \neq E$; and E' is an extension with the above three properties. As E is an extension, there is some D such that $E=Th(F \cup D)$. There must be some $\alpha \in D$ such that $\alpha \notin E'$, otherwise as $F \subseteq E'$ and $F \cup D \subseteq E'$ so $E=Th(F \cup D) \subseteq E'$. Now, "*assume* n:α"$\in \Delta$, $\neg \alpha \notin E'$ as $E' \subseteq E$ and $\neg \alpha \notin E$ (as $\alpha \in E$ and E is consistent). But $\alpha \notin E'$, contradicting 3.
Q.E.D.

3.3. Expressing Reiter's logic in our logic

Reiter's normal defaults of the form:

$$\frac{\alpha(\overline{X}):Mw(\overline{X})}{w(\overline{X})}$$

(where $\overline{X}$ is the set of free variables), can be approximated by the default

assume name$(\overline{X})$: $\alpha(\overline{X}) \supset w(\overline{X})$

Both of these can be used to explain $w(\bar{c})$, (for some ground terms $\bar{c}$) given $\alpha(\bar{c})$, as long as $w(\bar{c})$ is consistent. The second also allows us to explain $\neg \alpha(\bar{c})$ from $\neg w(\bar{c})$ as long as it is consistent, as well as being able to explain $\alpha(\bar{c}) \supset w(\bar{c})$. Notice that giving up the latter means giving up proof by cases. For example, from the following

$\alpha \vee \beta$
$\alpha : M\gamma / \gamma$
$\beta : M\gamma / \gamma$

we cannot derive γ in Reiter's system. If we allow proof by cases (with the law of the excluded middle) in Reiter's normal default logic we get exactly the logic above with a model theoretic semantics. All of the examples of Reiter[80] are still sensible when mapped to the form above.

It has been claimed that normal defaults are not adequate to handle most naturally occurring defaults (Reiter and Criscuolo[81], Etherington and Reiter[83]). Our logic provides another mechanism to handle these problems, namely the ability to prefer one theory over another. Poole[85] gives a mechanism to handle inheritance problems (Touretzky[84]) by preferring the most specific theory.

4. Diagnosis

Our definition of explainability also provides a natural characterisation of diagnosis. A diagnosis is a theory that implies some set of observations. Here we mean systems which do diagnosis from first principles, (e.g., Genesereth[84], Reiter[85]), rather than systems which encode the methodology of an expert, like rule-based expert systems such as Mycin (Buchanan and Shortliffe[84]).

With the same language and semantics, we let Δ be a set of possible malfunctions or diseases, as well as defaults describing how these problems normally manifest themselves and how the system normally operate. The theory is a detailed description of what is happening to produce the symptoms.

Theory D_1 is **simpler** than D_1 if $F \cup D_2 \supset D_1$. A **diagnosis** is a simplest theory (in the partial ordering of simplicity of theories) that explains the observations and is consistent. Notice that we have defined a semantic notion of simplicity. In particular, one theory is simpler than another if it is a subset of it. If a theory is a super set of another, then it implies the other, and is of no use, as it is only including hypotheses which were not needed to explain the observations. Thus by excluding larger theories from being diagnoses, we are just excluding theories which contain extraneous hypotheses. We are only using the hypotheses that we need to account for the observations.

We are also not adding heuristic notions of simplicity, for example, preferring theories which assume that fewer parts are faulty, or which maximise normality. Nor are we excluding diagnoses where someone may have some rare but unlikely disease. Which diagnosis is more likely, and which is more serious is a question of theory preference, and is not discussed here.

The idea of a diagnosis is that we are trying to determine reality (what is really the case) from our observations. The correctness and completeness of our semantics can be shown by how accurately our diagnoses corresponds to reality. The next theorem show that one of the diagnoses must correspond with reality if we have adequate knowledge, and adequate possible hypotheses to characterise the world we are modelling.

We say that our system is **adequate** to describe a situation (the relevant parts of the world we are describing) if there is some set of instances of hypotheses which are true in the world (correspond to "reality") and, together with the facts imply the observations.

Theorem 4: (diagnosis): If we have some symptoms and facts about a world, and our system is adequate to describe that world, and if $D_1, \cdots, D_n$ are the diagnoses then one of the D_i is correct (is true in the world).

Proof: Suppose W is the world, and F is true in W and there is some set δ of instances of possible hypotheses which are true in W and adequate to prove the observations. In this case δ is a theory, as $F \cup \delta \models O$ and $F \cup \delta$ is consistent (as W is a model). We need to show that some D_i is true in W. If no D_i is true in W, then a minimal subset of δ which predicts O, is a diagnosis (it is consistent at $F \cup \delta$ is consistent, and is finite as we only need the elements needed to prove O), and there are only a finite number of these. So one D_i must correspond with reality.
Q.E.D.

Whenever we are presenting a set of diagnoses, there will be the implicit assumption that we have enough knowledge to describe the world.

4.1. Testability

The main feature of a scientific theory or a diagnosis is its testability. If there are two possible diagnoses then we should be able to design tests which can discriminate amongst them.

If D_1 and D_2 are diagnoses, then we say that they are **distinguishable** if there is some proposition p such that

$$F \cup D_1 \supset p$$
$$F \cup D_2 \supset \neg p$$

That is D_1 predicts p and D_2 predicts $\neg p$.

We can **discriminate** between D_1 and D_2 by finding out whether p is true in the world (e.g., our implementation achieves this by asking the user).

Lemma 5: If D_1 and D_2 are diagnoses which can be distinguished by p then finding p true will remove D_2 and all theories less specific than D_2 (That is we are actually removing diagnoses by making tests).

Proof: By finding p true, we are adding p to the facts. $F \cup D_2 \supset \neg p$ so $F \cup \{p\} \cup D_2$ is inconsistent and cannot be a diagnosis. Any diagnosis less specific than D_2 implies D_2, and so is also inconsistent. Q.E.D.

Theorem 6: (testability): If $\{D_1, \cdots, D_n\}$ is the set of diagnoses, and the D_i are pairwise distinguishable, then there is a set of (at most $n-1$) tests which can determine which of the D_i are true in the actual world.

Proof: Suppose there are two distinguishable diagnoses D_i and D_j, then there is some test which will distinguish them. Carrying out this test will eliminate one of these. If there is not a pair of diagnoses left, then there is only one left. By the diagnosis theorem one of the D_i is consistent with reality. So it must be the one left. Q.E.D.

5. Examples

In this section we will give two different examples of diagnosis. One of these is in a rather contrived, pseudo-medical domain intended to show how we can handle diseases and symptoms. The second is in the domain of circuit diagnosis, which has been used by a number of other researchers, and so can be used to compare our system with theirs.

5.1. Example 2: A Pseudo-medical domain

Figure 2 shows an example session with Theorist.

---

facts and possible hypotheses

:**fact** aching(elbow) ← tennis-elbow;
:**fact** aching(hands) ← dishpan-hands;
:**fact** aching(X) ← arthritis joint(X);
:**fact** joint(elbow);
:**fact** joint(hands);
:**fact** joint(knee);
:**fact** plays-tennis ← tennis-elbow;
:**assume** patient-has-tennis-elbow: tennis-elbow;
:**assume** patient-has-dishpan-hands: dishpan-hands;
:**assume** patient-has-arthritis: arthritis;
:**assume** patient-has-meningitis: meningitis;
:**explain** aching(elbow) aching(hands); *%to be explained*

diagnoses

Theory: [patient-has-dishpan-hands,patient-has-tennis-elbow]
Theory: [patient-has-arthritis]

Make aching testable, and reask

:askable aching(X);
:**explain** aching(elbow) aching(hands);
Is aching(knee) true? *%system asks*
no; *%user replies*
Theory: [patient-has-dishpan-hands,patient-has-tennis-elbow] *%resulting theory*

Figure 2 Diagnosis in Theorist 0.1

The facts state the effects of diseases. For example, the first fact states that if a person has tennis elbow, then they have an aching elbow. We describe the diseases as possible hypotheses. A diagnosis is then a minimal set of diseases which is consistent, and predicts the observations.

The system comes up with two diagnoses; namely that the patient has both dishpan hands and tennis elbow, and the diagnosis that the patient has arthritis. If we subsequently assert that the system can ask if some part is aching (i.e., it can be used as part of a test), then answering that the person does not also have an aching knee rules out the second diagnosis.

Here we have described the effects of diseases as absolute. For example, the third fact says that all joints ache if one has arthritis. If this is not a fact, but a default, then it can be described as:

assume aching_by_arthritis(X): *aching*$(X) \leftarrow$ *arthritis* $\wedge$ *joint*(X);

Thus we can use arthritis to explain any aching joint, but we are not forced to conclude that all joints ache just because someone has arthritis. This allows us to have defaults and diagnosis in the same system, with a uniform and simple semantics.

5.2. Circuit Diagnosis

The second example we will use is the diagnosis of faults in digital circuits. Consider the full adder circuit of figure 3.

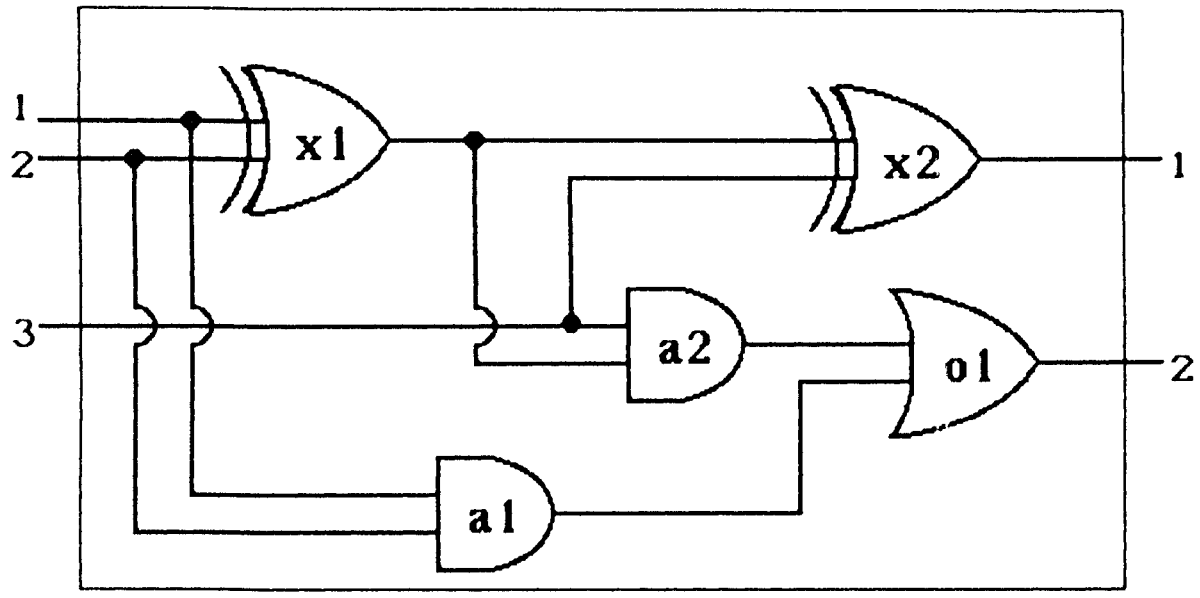

Figure 3 A Circuit

The representation of this, is similar to that given in Genesereth[84]. We have rules about how faulty gates act, as well as how non-faulty gates work and how the gates are connected. The possible hypotheses are that some gates are working properly and some gates are faulty. Here we describe two different diagnostic systems based on Theorist. The first is a static system which can diagnose a circuit fault when provided with input and output values. The second is based on a theory of how circuits actually fail (Chang et.al.[70]), and accepts a set of input/output pairs to perform the diagnosis.

5.2.1. Static Circuit Diagnostic System

```
fact gate(x1 xor);
fact gate(x2 xor);
fact gate(a1 and);
fact gate(a2 and);
fact gate(o1 or);

fact conn(in(1,f1),in(1,x1));
fact conn(in(1,f1),in(1,a1));
fact conn(in(2,f1),in(2,x1));
fact conn(in(2,f1),in(2,a1));
fact conn(in(3,f1),in(2,x2));
fact conn(in(3,f1),in(1,a2));
fact conn(out(1,x1),in(1,x2));
fact conn(out(1,x1),in(2,a2));
fact conn(out(1,a1),in(2,o1));
fact conn(out(1,a2),in(1,o1));
fact conn(out(1,x2),out(1,f1));
fact conn(out(1,o1),out(2,f1));
```

Figure 4 A Definition of the Circuit of Figure 3

Figure 4 describes the circuit in figure 3. *gate*(D,T) says that device D is a gate which is supposed to be of type T. *in*(N,D) where N is a number, and D device is the Nth input to device D. *conn*($W1$,$W2$) means that input (or output) $W1$ is connected by a wire to device $W2$. The total circuit is known as device $f1$.

Figure 5 gives a description of how gates are intended to work. The assertion

```
ttable(T,I1,I2,O)
```

means that a gate of type T, with inputs $I1$ and $I2$ produces output O. The value "*anything*" means that the value of that input is irrelevant.

```
fact ttable(and on on on);
fact ttable(and off anything off);
fact ttable(and anything off off);
fact ttable(or off off off);
fact ttable(or on anything on);
fact ttable(or anything on on);
fact ttable(xor off on on);
fact ttable(xor off off off);
fact ttable(xor on on off);
fact ttable(xor on off on);
```

Figure 5 A description of how gates are intended to work

The first diagnosis program is given in figure 6. The first rule says that any input can have the value of "*anything*". The second says that a gate which is not faulty has the output which corresponds to the type of the gate according to the truth table. The third rule says that gates which are faulty produce the wrong output. The last rule for *val* says that the value of a connection is equal to the value of the connection it is connected to.

Note that the intended interpretation for $faulty(D)$ is that device D is faulty for the input that it is given at this time. There is no assumption that faulty gates always produce the wrong output. The facts say nothing about how the gate works for other inputs.

There are two possible hypotheses, namely that a gate is faulty, and that a gate is not faulty.

```
assume atFault(Device): faulty (Device );
assume ok(Device): ¬faulty(Device);
```

5.2.2. Example 3

If we provide a set of inputs, for example, if we tell the system that all of the inputs are off, and ask to explain the outputs being on, then the system must explain:

$$val(in(1,f1),off) \wedge val(in(2,f1),off) \wedge val(in(3,f2),off) \supset$$
$$val(out(1,f1),on) \wedge val(out(2,f1),on)$$

Note that this is posed by specifying $val(out(1,f1),on) \wedge val(out(2,f2),on)$ as a goal, and making $val(in(N,f1),V)$ askable.

The following table shows the 6 diagnoses found. Each row corresponds to a diagnosis. Each column says whether the corresponding gate is assumed to be atFault, or OK, or whether nothing need be assumed about that gate.

```
rule val (in(N Device) anything);

rule val(out(1 Device) Out1) ←
            gate (Device Type) ∧
            ¬ faulty (Device) ∧
            ttable(Type In1 In2 Out1) ∧
            val (in(1 Device) In1) ∧
            val (in(2 Device) In2);

rule val(out(1 Device) Out1) ←
            gate (Device Type) ∧
            faulty (Device) ∧
            ttable(Type In1 In2 Eout) ∧
            opp(Eout Out1) ∧
            val (in(1 Device) In1) ∧
            val (in(2 Device) In2);

rule val (Y Z) ←
            different(Z anything) ∧
            conn (X Y) ∧
            val (X Z);

fact opp(on off);
fact opp(off on);
```

Figure 6 Simple Circuit Diagnosis in Theorist

	a1	a2	o1	x1	x2
1		atFault	ok	atFault	ok
2		atFault	ok	ok	atFault
3	atFault		ok	atFault	ok
4	atFault		ok	ok	atFault
5	ok	ok	atFault	atFault	ok
6	ok	ok	atFault	ok	atFault

Diagnosis 1 corresponds to the theory

$[atFault(a2), ok(o1), atFault(x1), ok(x2)]$

This diagnosis says nothing about gate $a1$. The output of gate $a1$ is irrelevant to the output.

Because our diagnoses are covering (by the diagnosis theorem), we know that one of the diagnoses is correct. Thus, in particular we know:

$$atFault(x1) \vee atFault(x2)$$
$$atFault(a1) \vee atFault(a2) \vee atFault(o1)$$

This set of diagnoses also tells us more. It also tells us what other conditions are to hold. For example, in the case that $a1$ is at fault, we also know that $o1$ is producing the correct output, and that the value of $a2$ is irrelevant.

With further information we can discriminate between these theories, for example by probing internal values. Theories 1, 3 and 5 predict that the output of gate $x1$ is on, and the theories 2, 4 and 6 predict the output is off. By testing this output, we can eliminate three of the diagnoses.

Theories 3 and 4 predict the output of gate $a1$ is on, theories 5 and 6 predict the output of gate $a1$ is off, whereas theories 1 and 2 say nothing about the value of this output. The value of output $a1$ cannot be used to eliminate theories 1 and 2.

Note that we have not eliminated the possibility that gates $a1$ and $a2$ are both faulty. This indeed may be the problem.

5.3. Multiple Observations

One problem with the above diagnosis program is that it does not work with multiple input-output observations. This may not be a problem for many diagnostic tasks where we cannot vary the input, but can only make more observations about the system.

For circuits we can vary the inputs, and use the information of the output in a diagnosis. To use multiple observations we need to consider how circuits actually break down. First, we assume that the gates are consistent. That is, each gate produces the same output for a particular input.

We could assume that gates break down in arbitrary ways, and that we are trying to find the truth table for each gate. Our system could be extended to do this by adding an extra argument to the val predicate (the observation), and having the defaults $ok(D,In1,In2)$ mean that gate D works ok for the inputs $In1$ and $In2$, and having $atFault(D,In1,In2)$ mean gate D is faulty when the inputs are $In1$ and $In2$. We also cannot use the *anything* value, as the actual value may be significant. Such a diagnosis is combinatorially explosive, as we are not using any knowledge of how gates fail.

An alternative strategy is to make the single fault assumption (Genesereth[84]), or the minimal fault assumption (for the example above, we would assume only two faults). Note that this is not generally true, and may remove some diagnoses which may indeed be correct.

Our favoured alternative is to use a theory on how gates fail in practice (Chang et.al.[70]). According to this theory gates fail by either

1. outputs being stuck on or off or
2. inputs being stuck on or off.

To add this assumption to our system, we add an extra argument to the *val* predicate, which is the time (input-output observations). We also have a status for each gate. Instead of having a gate either *faulty* or *ok*, we give the gate a status which is one of the following:

ok - the gate is working properly
stuckon - the output is stuck on
stuckoff - the output is stuck off
*stuck*1 - input 1 is stuck to make the gate act like a wire from input 2 (that is input 1 is stuck on for and-gates, off for or-gates or off for xor-gates.)
*stuck*2 - input 2 is stuck
*stuck*1*on* - input 1 is stuck on for an xor-gate (that is the gate acts as an inverter for input 2)
*stuck*2*on* - input 2 is stuck on for an xor-gate.

```
fact fault(stuckon X anything anything on);
fact fault(stuckoff X anything anything off);
fact fault(stuck1 X anything on on);
fact fault(stuck1 X anything off off);
fact fault(stuck2 X on anything on);
fact fault(stuck2 X off anything off);
rule fault(stuck1on xor anything V1 V2) ← opp(V1 V2);
rule fault(stuck2on xor V1 anything V2) ← opp(V1 V2);
```

Figure 8 Faults for stuck gates

Figure 8 shows how faulty gates work.

fault(*Prob*,*Type*,*In*1,*In*2,*Out*)

means a gate of type *Type*, with problem *Prob*, produces output *Out* from inputs *In*1 and *In*2.

Figure 9 gives the modified rules for the new val that allows for multiple observations. The first two rules correspond to the first two in Figure 6. The third says that gates have the problems as in the fault table. The last rule says that a gate can only have one status.

5.3.1. Example 3

Suppose that the system is given the following observations to explain

```
rule val (in(N Device) anything T);

rule val(out(1 Device) Out1 T) ←
            gate (Device Type) ∧
            status(Device ok) ∧
            ttable(Type In1 In2 Out1) ∧
            val (in(1 Device) In1 T) ∧
            val (in(2 Device) In2 T);

rule val(out(1 Device) Out1 T) ←
            gate (Device Type) ∧
            fault(St Type In1 In2 Out1) ∧
            status(Device St) ∧
            val (in(1 Device) In1 T) ∧
            val (in(2 Device) In2 T);

rule ¬status(Device Status) ←
      status(Device St) ∧
      different(Status St);
```

Figure 9 Diagnosis System based on gates being stuck

Time	$in(1,f1)$	$in(2,f1)$	$in(3,f1)$	$out(1,f1)$	$out(2,f1)$
t1	on	on	on	on	on
t2	off	on	off	off	off
t3	on	off	off	on	off
t4	on	on	off	off	on

The following four diagnoses are generated:

	a1	a2	o1	x1	x2
1	ok	stuck2	ok	stuck1on	ok
2	ok	ok	ok	stuck1on	ok
3	ok	stuckoff	ok	stuck1on	ok
4	ok		stuck1	stuck1on	ok

From the diagnosis theorem, if the gates are acting in accordance to our assumptions, then we know:

1. x2 is OK
2. x1 is stuck1on

3. a1 is OK.

6. A Comparison with Other Systems

6.1. Genesereth's DART

A DART diagnosis is a disjunction of faulty gates (Genesereth[84]). For example, from the observations in example 3, the diagnoses are:

$$faulty(x1) \vee faulty(x2)$$
$$faulty(a1) \vee faulty(a2) \vee faulty(o1)$$

This can be implemented in Theorist by using

assume ok(D): $\neg$faulty(D);

as the only possible hypothesis. We only need rules which state how non-faulty gates work and try to prove that some gate is faulty. A diagnosis takes the form

$$\neg faulty(g1) \wedge \neg faulty(g2) \wedge \cdots \wedge \neg faulty(gn)$$

which is consistent and minimal and implies $faulty(g0)$. If this is the case, then we know

$$faulty(g0) \vee faulty(g1) \vee \cdots \vee faulty(gn)$$

and this is a minimal faulty disjunction (as the theory is minimal, and consistent).

If we compare DART's diagnosis with that given in example 2, we find that Theorist's diagnosis implies the diagnosis produced by DART. It also provides more information, namely what other information we are assuming (i.e., in the assumption that $a1$ is faulty).

The advantage is that Theorist can use knowledge about how faults manifest themselves. DART can only use knowledge about how properly working systems are intended to work. This may be appropriate for some domains, but we can use both sorts of knowledge.

6.2. Reiter's Theory of Diagnosis

Reiter[85] defines a diagnosis as a minimal set of faulty assumptions which are consistent with the observations. Anything not assumed to be faulty is assumed to be OK.

Example 3 in Reiter's Theory produces the following diagnosis

	a1	a2	o1	x1	x2
1	ok	atFault	ok	atFault	ok
2	ok	atFault	ok	ok	atFault
3	atFault	ok	ok	atFault	ok
4	atFault	ok	ok	ok	atFault
5	ok	ok	atFault	atFault	ok
6	ok	ok	atFault	ok	atFault

Note that this is very similar to our diagnosis, but requires assumptions about the operation of all gates, even if they are irrelevant to the observations.

We have avoided the problems that arise with existential and universally quantified variables in defaults (Poole[84]) by requiring that we only use ground instances of defaults in theories.

8. Conclusion

We have shown how the use of theory formation, using constrained forms of hypotheses, fits the intuition of default reasoning and can be used as the basis for diagnostic systems.

One of the remaining questions concerns the comparison of theories. That is, determining when one theory is "better" than another. Poole[85] argues that the problem of inheritance in semantic nets can be best done by preferring the most specific theory. That is, when there is specific knowledge and more general knowledge available, then we prefer to use the most specific knowledge. Jones and Poole[85] describes how we can get a diagnosis at the appropriate level of detail in the domain of educational diagnosis. For a learning system we may prefer the most general theory, and for a domain with uncertainty, we may prefer the most probable theory, or the most serious (the one that, if it were the problem, would require immediate therapy).

This theory provides the basis for the Theorist system, which we are using for a variety of domains, including circuit diagnosis, and diagnosis of students with learning disabilities.

Acknowledgements

This work was started while the author was at the Australian National University. Special thanks to my supervisor, Robin Stanton of the Australian National University. Thanks also to Hugh MacKenzie and Peter Creasy for many discussions and Randy Goebel, Eric Neufeld and Paul Van Arragon for valuable comments on an earlier draft of this paper.

References

Buchanan and Shortliffe[84], Rule-Based Expert Systems.

Chang,H.Y., Manning,E.G. and Metze,G.[70], *Fault Diagnosis of Digital Systems*, Wiley-Interscience, 1970.

Doyle,J. [79], "A Truth Maintenance System", *Artificial Intelligence* Vol 12, pp 231-273.

Etherington,D and Reiter,R.[83], "On Inheritance Hierarchies with Exceptions", *Proc. AAAI-83*, pp. 104-108.

Genesereth,M.R.[85], "The use of design descriptions in automated diagnosis", *Artificial Intelligence 24*, pp.411-436.

McCarthy,J.[80], "Circumscription - A form of non-monotonic reasoning", *Artificial Intelligence, 13*, pp.27-39.

McCarthy,J.[84], "Applications of Circumscription to Formalising Common Sense Knowledge", *Proc. AAAI Workshop on Non-monotonic Reasoning*, NY, Oct, 295-324.

The diagnosis theorem is not a theorem of Reiter's theory. One of his diagnoses does not necessarily correspond to reality. For example, the problem that gates $a1$, $a2$ and $x1$ were producing the wrong results for this input (which would produce the observed input/output pairs), is not covered by his diagnoses. Rediagnosis is required for any extra information (for example, if a probe was added to find internal values).

6.3. Set Covering Model

Reggia, Nau and Wang[83] propose a theory of diagnosis based on a set covering model. This is based on having a causal relation between diseases and manifestations, and finding a minimal (i.e. minimal cardinality) set of diseases which cover the observed symptoms.

This can be implemented in Theorist by having diseases, and implications of the form "$disease \supset manifestation$" as possible hypotheses. This fits in with the intuition in Reggia at.al.[83], where the intended interpretation is that the manifestation may occur if the disease is present. A diagnosis (in our sense) becomes a minimal set of diseases and default manifestations which imply the observations.

Note that this system, unlike DART, only allows knowledge of how diseases manifest themselves, and does not allow "deeper" reasoning about how the system is intended to work normally. There is also no assumption that other components are not faulty. The diagnosis is not refutable, as each manifestation is only a default. Symptoms which always occur with a disease (eg., mercury poisoning implies the presence of mercury in the blood) cannot be stated in this system.

Note that the definition if minimal means "minimal cardinality", rather than set inclusion. This means that the diagnosis theorem is not a property of their system (for example, the patient actually has 3 diseases which are all needed to explain the observations, even though there are 2 diseases which also explain them).

7. Implementation

The implementation of Theorist is very simple, is closely related to that of Reiter[80], and is described elsewhere (Poole, Goebel and Aleliunas[85]).

We attempt to prove the observations with the facts and possible hypotheses as axioms. We make D the set of instances of possible hypotheses used in the proof. We then know,

$$F \cup D \models O$$

We then use a complete theorem prover to prove that D is consistent, by failing to prove that D is inconsistent. In general this is undecidable, but has not been a problem in the domains we have considered.

There is currently an interpreter (Poole, Goebel and Aleliunas[85]), based on a complete theorem prover written in Prolog (Poole and Goebel[85]). There is also a compiler which compiles Theorist facts, rules and defaults into Prolog. A more efficient system based on dependency directed backtracking (e.g., Doyle[79]) is currently under development.

McDermott,D.V. and Doyle,J.[80], "Non Monotonic Logic 1", *Artificial Intelligence,* Vol 13, pp 41-72.

Moore,R.C. [83], "Semantical Considerations on Nonmonotonic Logic", *IJCAI-83* pp 272-279.

Poole,D.L. [84], "A Logical System for Default Reasoning", *AAAI workshop on Non-monotonic Reasoning,* NY Oct 1984, pp 373-384.

Poole,D.L., Goebel,R.G. and Aleliunas,R.[85], "Theorist: a logical reasoning system for defaults and diagnosis", invited to appear in N. Cercone and G.McCalla (Eds.) *Knowledge Representation.*

Poole,D.L.[85] "On the Comparison of Theories: Preferring the Most Specific Explanation", *Proc. IJCAI-85,* pp.144-147.

Popper,K.R. [59], *The Logic of Scientific Discovery,* Basic Books, New York.

Quine,W.V. and Ullian,J.S.[78], *The Web of Belief,* Random House, Yew York, Second Edition.

Reggia,J.A., Nau,D.S. and Wang,P.Y.[83] "Diagnostic expert systems based on a set covering model", *Int. J. Man-Machine Studies 19,* pp.437-460.

Reiter,R. [80], "A Logic for Default Reasoning", *Artificial Intelligence,* Vol 13, pp 81-132.

Reiter,R. and Criscuolo,G.[81], "On Interacting Defaults", *Proc. IJCAI-81,* pp.270-276.

Reiter,R.[85], *A Theory of Diagnosis from First Principles.*

Seki,H. and Takwuchi,A.[85], *An Algorithm for Finding a Query That Discriminates Competing Hypotheses,* ICOT Research Paper.

Touretzky,D.S.[84], "Implicit Ordering of Defaults in Inheritance Systems", *Proc AAAI-84,* pp 322-325.