Fixed Predicates in Default Reasoning

David Poole

Logic Programming and
Artificial Intelligence Group

Research Report CS-87-11
February 1987

Fixed Predicates in Default Reasoning

David Poole
Logic Programming and Artificial Intelligence Group,
University of Waterloo,
Waterloo, Ontario, Canada, N2L3G1
(519) 888-4443
dlpoole@waterloo.csnet

February 23, 1987

Abstract

One of the interesting differences between Circumscription and other proposals for formalising nonmonotonic reasoning is the ability to partition predicates into varying and fixed predicates. In this paper we show how corresponding distinctions can be added to default reasoning systems. A way to implement this, as well as some potential applications are discussed.

science track
topic area: automated commonsense reasoning
keywords: default reasoning, circumscription, variable and fixed predicates

1 Introduction

Circumscription [McCarthy86,Lifschitz86] has one interesting feature which is not shared by other systems for non-monotonic reasoning. This is the ability to distinguish between relations which are able to vary when making assumptions about other relations, and those which cannot.

As shown by [Etherington85] this distinction is essential to circumscription. If no variables are allowed to vary, then circumscription can only

change the relation being minimised. This is not very interesting from the point of view of finding the consequences of one's assumptions.

The following example shows what varying and non-varying means.

Example 1 Let our default be "ontable". Let the only other thing given as an axiom be

$$red \Rightarrow \neg ontable$$

Intuitively we want this to mean that a block is on the table by default, however if it is red then it is not on the table.

In each of [Reiter80,Poole86,Moore84] we can "deduce" *ontable*. *ontable* can also be proven from the circumscription [McCarthy86] with *red* allowed to vary. In each of these, *ontable* is consistent with all of the facts, and so can be assumed. Once we have *ontable*, we can conclude $\neg red$.

The problem with this is that the side effect of concluding something about the redness of the block may not be desired. We may not want to assume anything about the redness of the block. An alternate answer to the question of whether the block is on the table is "yes, unless it is red".

If one circumscribes $red \Rightarrow \neg ontable$, with red fixed (not allowed to vary) then one does derive $ontable \equiv \neg red$. Note also that ontable does not follow from the circumscribed formula (in some sense we still need some form of conditional answers as suggested by this paper).

Example 2 Let us suppose we had no defaults and the fact

$$\neg red \Rightarrow ontable$$

This says that a block is on the table if it is not red, and says nothing about the case if a block is red. From this, we cannot conclude *ontable*.

These two examples are interesting if we are allowed to reason by cases. Consider the two cases of the block being red and the block being not red. In each of these cases the second example provides stronger support for *ontable* being true than the first example. If *red* is false, then the second says that *ontable* is true; the first says it is true by default. If *red* is true, the first says that *ontable* is false; the second says nothing about whether it is true or false.

The difference is when we do not know whether the block is red or not. In the first case, it allows us to assume that it is on the table (and so not red), and in the second we are not allowed to assume anything.

The problem is how to handle the side effects of making assumptions. It would be nice to only make these side effects explicitly, and to be allowed to control implicit side effects of making assumptions. This papers shows how this can be done effectively in a default reasoning setting.

2 Formal Framework

This work is placed in the context of Theorist [Poole86,PGA86], but seems to be applicable to all systems which use consistency or failure to prove as a basis for concluding things.

We use the standard syntax of the first order predicate calculus, with variables in upper case.

F is a set of closed formulae (called *facts*), which we are giving as true

Δ is a set of formulae, from which instances can be used as a possible hypotheses[1].

Definition 1 We say formula g is *explainable* if there is some D, a set of instances of elements of Δ, such that

$$F \cup D \models g$$

$$F \cup D \text{ is consistent}$$

D is said to be the theory that explains g.

3 Fixed Predicates

In this section we introduce the notion of *fixed predicates*. These should be interpreted as those predicates about which we do not want to be able to make implicit assumptions. For example, in the example above, we

[1] $w \in \Delta$ is equivalent to [Reiter80]'s normal default : Mw/w [Poole86].

could make *red* fixed if we did not want the system to be able to make an assumption (as a side effect of the defaults) of the block being red. Let Φ be the set of fixed predicates.

Definition 2 g is *conditionally explainable* from F, Δ and Φ, if there is a set D of instances of elements of Δ, and a formula C made of instances of elements of Φ (under conjunction, disjunction, negation), such that

1. $F \wedge C \wedge D \models g$
2. $F \wedge C \wedge D$ is consistent
3. if $F \wedge C \wedge D \models \phi$, where ϕ is a formula made from elements of Φ, then $F \wedge C \models \phi$.

D is said to be the theory that explains g, and C is the condition for D.

Example 3 consider
$F = \{red \Rightarrow \neg ontable, onfloor \Rightarrow \neg ontable\}$,
$\Delta = \{ontable\}$ and
$\Phi = \{red\}$.
The intended interpretation is that we may assume that the block is on the table, but we cannot make any assumptions about the redness of the block. Here *ontable* is conditionally explainable with theory $\{ontable\}$ and condition $\{\neg red\}$. This is to be read that the block is on the table if it is not red. We have allowed the side effect that $onfloor$ is affected by the assumption.

Example 4 Let

$$\Delta = \{ontable, redonfloor\}$$

$$F = \{\ red \Rightarrow \neg ontable,\ red \wedge redonfloor \Rightarrow onfloor,\ onfloor \Rightarrow down,\ ontable \Rightarrow down\}$$

$$\Phi = \{red\}$$

We can conditionally explain *ontable* with the theory $\{ontable\}$ and the condition $\neg red$. We can conditionally explain $onfloor$ with the theory $\{redonfloor\}$ and the condition *red*.

Intuitively, we should have more confidence in *down* as it can be explained independently of whether the block is red or not. If it is red, then it is on the floor; if it is not red then it is on the table. In either case it is *down*. The following definition formalises this intuition.

Definition 3 g is *unconditionally explainable* from F, Δ, Φ if g is conditionally explainable with theories D_i and corresponding conditions C_i for $i = 1, n$, such that $F \models C_1 \vee ... \vee C_n$.

Example 5 In the previous example, *down* is unconditionally explainable. $D_1 = \{ontable\}$, $C_1 = \neg red$, $D_2 = \{redonfloor\}$, $C_2 = red$.

The following example shows that a goal can be unconditionally explainable (definition 2), yet not explainable (definition 1).

Example 6 Let $\Delta = \{ontable, onfloor\}$,
$F = \{red \wedge ontable \Rightarrow g, \neg red \wedge onfloor \Rightarrow g, ontable \Rightarrow \neg onfloor\}$,
$\Phi = \{red\}$.
In this case we can unconditionally explain g, with $D_1 = \{ontable\}$, $C_1 = red$, $D_2 = \{onfloor\}$, $C_2 = \neg red$. g is not explainable, as the only potentially explainable theory is $\{ontable, onfloor\}$, which is not consistent.

The preceding examples shows the intuition behind the fixed predicates. They are intended to be predicates which can subsequently (and independently) be shown to be true or false. We want to ensure that an answer can be found independently of whether they turn out to be true or false.

This means that some relations should not be fixed, for example *onfloor* should not be fixed with *ontable* allowed to vary, as these are not independent conditions. *ontable* and *red* may be independent, in so far as we may find out later whether the block is indeed red, and we want to reason about whether the block is on the table independently of whether the block is red or not.

Possible applications of this may include

- in planning we may want to have a plan depending on some condition which is not known at the time, but which can be determined when the plan is executed. We would like the plan to work whether the condition turns out to be true or not.

- in decision making where we do not want to make a condition implicit upon some condition which we cannot determine. We may want to make sure our decision is appropriate whether or not some person is telling the truth. In this case we want to be able to explain our decision independently of whether the person in honest. This is done by making honesty fixed, but unknown.
- in a diagnostic setting, it may be appropriate to make the values of some tests fixed. If we can explain the same results independently of the value of a test, then it is not appropriate to carry out the test. If different diagnoses are conditionally explained, then the it is appropriate to carry out the tests defining the conditions.

4 Comparison to other systems

The only other system which makes a distinction similar to the one in this paper is Circumscription [McCarthy86,Lifschitz86].

The underlying logic in this paper is similar to other systems which rely on consistency or failure to prove as the basis for conclusions. Elements δ of Δ correspond to Reiter's [Reiter80] normal defaults of the form : $M\delta/\delta$ [Poole86], and also seem to correspond to $\neg L\neg\delta \Rightarrow \delta$ in [Moore84]. These systems, however, do not make the distinctions made in this paper.

The definition given here for conditional explainability does not exactly correspond to circumscription (minimising the negations of the elements of Δ). Consider example 6, where we unconditionally explain g. If we circumscribe $\{\neg ontable, \neg onfloor\}$ with red fixed, then we just add the formula $ontable \vee onfloor$ to the axioms, from which we still cannot conclude g.

The difference is that we treat fixed predicates as relations for which we do not know the truth values, and for which we may not assume that they have any values as a side effect of other assumptions. We can unconditionally explain some goal if we can explain the goal no matter what the values of the fixed predicates are. We are assuming that the values are implicitly knowable. We may not know which theory is appropriate until we know the values. Circumscription, however treats fixed variables as somehow *unknowable*. We must be able to give a theory independently of the values of

the variables. This seems to correspond to the following definition:

Definition 4 g is *consistently unconditionally explainable* if g is unconditionally explainable with theories $D_1, ..., D_n$ such that $D_1 \wedge ... \wedge D_n$ is consistent with F.

In this case, if we have $D = D_1 \wedge ... \wedge D_n$, then $F \wedge D \models g$ and if $F \wedge D \models \gamma$ then $F \models \gamma$. That is g is explainable from theory D which does not affect the truth of values of γ. A detailed comparison with Circumscription is beyond the scope of this paper.

5 Implementation

Note that the notion of explainability for the predicate calculus is undecidable (not even recursively enumerable). The following is argued in terms of a theorem prover which halts. If the system does not halt, then we cannot conclude explainability or not. The procedure will be given in terms of a complete backward chaining theorem prover (for example linear resolution [Chang73], which is complete in the sense that a goal can be proven if it follows from a consistent set of axioms).

Essentially the "algorithm" follows the definition of explainability. To unconditionally explain g from F, Δ, Γ, try to prove g from $F \cup \Delta \cup \Gamma$. Make D the set of instances of elements of Δ used in the proof. Make C_0 the set of (positive and negative) instances of elements of Γ which cannot be proven from F. If we find a proof then we know

$$F \cup D \cup C_0 \models g$$

We can check conditions (2) and (3) of definition 2 for conditional explainability, by trying to prove $\neg(D \wedge C_0)$ from F together with Γ. Let C_i be the conjunction of instances of elements of Γ used in each proof of $\neg(D \wedge C_0)$ which could not be proven from F. If one $C_i = \{\}$, then $F \cup D \cup C_0$ is inconsistent. Otherwise we have $F \wedge C_i \models \neg(D \wedge C_0)$, that is $F \wedge D \wedge C_0 \models \neg C_i$ for all i.

We now have g is conditionally explainable from theory D and condition $C = C_0 \wedge \bigwedge_i \neg C_i$. We know $F \wedge D \wedge C$ is consistent as $F \wedge D \wedge C_0$ is consistent (as a complete theorem prover failed to prove it is inconsistent),

and it implies $F \wedge D \wedge C$. Property 3 of definition 2 holds by virtue of how we constructed C. Essentially we added all of the elements of Γ which $F \wedge D \wedge C$ implied, to C.

To compute unconditional explainability of g, we now try to explain g from $F \wedge \neg C$, if it is consistent. If it is inconsistent, then we have $F \models C$, and so g is unconditionally explainable; otherwise we must cover the other cases, until the generated conditions cover all cases.

6 Conclusion

We have shown how the notion of fixed predicates may be defined for Default reasoning. A few examples of how this may be useful are suggested (but, of course, only experience will tell us how useful this notion is). An implementation is outlined which is not very much more complicated than the implementation for default reasoning [PGA86,Poole87] (the problem is undecidable, so it is hard to give a comparison).

The most valuable contribution of this paper is probably in that it provides one more piece of the jigsaw puzzle to give a comparison of two of the major proposals for formalising non-monotonic reasoning, namely default reasoning and Circumscription. A detailed comparison is beyond the scope of this paper.

Acknowledgements

This research was supported under NSERC grant A6260. Thanks to Eric Neufeld, Paul Van Arragon, Denis Gagné, Bruce Kirby and Scott Goodwin for valuable discussions, arguments and comments on an earlier draft of this paper.

References

[Chang73] C. Chang and R. Lee, *Symbolic Logic and Mechanical Theorem Proving*, Academic Press, 1973.

[Etherington85] D. Etherington, R. Mercer and R. Reiter, "On the Adequacy of Predicate Circumscription for Closed-World Reasoning", *Computational Intelligence 1*, (1985) 11-15.

[Lifschitz86] V. Lifschitz, "Pointwise Circumscription: Preliminary Report", *Proc. AAAI-86*, pp.406-410.

[McCarthy86] J. McCarthy, "Applications of Circumscription to formalizing commonsense reasoning", *Artificial Intelligence 28*, 1986, 89-118.

[Moore84] R.C. Moore, "Semantical Considerations of Nonmonotonic Logic", *Artificial Intelligence 25*, 1984, 75-94.

[Poole85] D. Poole, "On the Comparison of Theories: Preferring the Most Specific Explanation", *Proceedings Ninth International Joint Conference on Artificial Intelligence*, Los Angeles, August 1985, pp. 144-147.

[Poole86] D. Poole, *Default Reasoning and Diagnosis as Theory Formation*, Technical Report CS-86-08, Department of Computer Science, University of Waterloo, March 1986, 19 pages.

[Poole87] D. Poole, *Building Consistent Theories*, Technical Report, Department of Computer Science, University of Waterloo, February 1987.

[PGA86] D. Poole, R. Goebel and R. Aleluinas, "Theorist: a logical reasoning system for defaults and diagnosis", in N.Cercone and G.McCalla (Eds.) *The Knowledge Frontier: Essays in the Representation of Knowledge*, pp. 331-352.

[Reiter80] R. Reiter, "A Logic for Default Reasoning", *Artificial Intelligence*, Vol 13, pp. 81–132.