Indefinite Reasoning with Definite Rules

L. Thorne McCarty and Ron van der Meyden
Computer Science Department
Rutgers University
New Brunswick, NJ 08903, USA
mccarty@cs.rutgers.edu, meyden@paul.rutgers.edu

Abstract

In this paper we present a novel explanation of the source of indefinite information in common sense reasoning: Indefinite information arises from reports about the world expressed in terms of concepts that have been defined using only definite rules. Adopting this point of view, we show that first-order logic is insufficiently expressive to handle important examples of common sense reasoning. As a remedy, we propose the use of circumscribed definite rules, and we then investigate the proof theory of this more expressive framework. We consider two approaches: First, *prototypical proofs,* a special type of proof by induction, which yields a sound proof theory. Second, we describe cases in which there exists a *decision procedure* for answering queries, a particularly significant result because it shows that it is possible to have decidable query processing in circumscribed theories that are not equivalent to any first order theory.

1 Introduction

Common sense reasoning has a bias towards *definite information,* as many researchers have noted. Thus Johnson-Laird [1983] has argued that human beings draw inferences in particular "mental models'', and Levesque [1986] has urged, for similar reasons, the study of "vivid knowledge'' in artificial intelligence. This bias towards definite information has also permeated computer science proper. The field of logic programming, for example, is based on a subset of first-order logic consisting entirely of definite clauses. Indeed, Makowsky [1985] has argued that "Horn formulas matter" in computer science precisely because they have a unique minimal model.

If this is so, then how does *indefinite information* arise in common sense reasoning? In this paper, we adopt a novel point of view on this question, and explore the consequences. We suggest (as a Gedanken experiment) that common sense reasoning is based exclusively on *definite rules,* and that indefinite information arises from the application of these definite rules to definite facts in situations where communication is almost always incomplete. An individual who receives reports about the world does not usually know the definite facts upon which these reports are based, and for such an individual the world is a (chaotically!) indefinite place. Out of necessity, an individual in this situation does indefinite reasoning with definite rules.

If we adopt this point of view, however, it turns out that the standard way of representing indefinite information in artificial intelligence - i.e., by a first-order language capable of asserting disjunctive and existential facts - is inadequate. We show in Section 3 that first-order logic cannot express a simple form of indefinite information that seems essential for common sense reasoning. Instead, we suggest in Section 4 that an adequate representation of indefinite information requires the use of *circumscription* [McCarthy, 1980; McCarthy, 1986]. But circumscription is a second-order formalism. Is it plausible to propose such a complex logical language for such a mundane purpose as common sense reasoning? We believe the answer is: Yes. The principal technical contribution of the present paper, in Sections 5 and 6, is to suggest two new techniques for computing the inferences that follow from a circumscribed definite rulebase. These techniques apply in special cases, of course, but they seem promising for various practical applications.

In Section 5, we develop the notion of a *prototypical proof*, which is a special type of inductive proof. If a query is entailed by a circumscribed rulebase, then a prototypical proof is guaranteed to exist, and if a prototypical proof succeeds, then an induction schema can be automatically generated. Furthermore, an induction schema in our system has very simple and appealing proof-theoretic properties. This means that we can search systematically for inductive proofs, at least in certain special cases that seem to be of considerable practical importance. These techniques cannot give us a complete proof procedure, of course. But in Section 6, we show that an actual *decision procedure* exists for certain other special cases that seem to be of practical importance. Previous work on circumscription has tried to find conditions under which a circumscribed theory "collapses" to a first-order theory [Lifschitz, 1985], so that standard theorem-provers for first-order logic could then be applied. However, these conditions are extremely re-

strictive, and recent work by Kolaitis and Papadimitriou [1990] has shown that the existence of an equivalent first-order theory is itself an undecidable property. Our results in Section 6 improve upon this previous work in two ways: (i) we show that complete query-answering is possible even for theories whose circumscription is not first-order; and (ii) we provide a full decision procedure in such cases rather than a semi-decidable proof theory.

The overall framework in which we have conducted this research is presented in Section 2, which is based on previously published work [McCarty, 1988a; McCarty, 1988b; Bonner *et al.,* 1989]. In Section 2, we define precisely what we mean by a "definite" rule, and we argue that our logic should be *intuitionistic,* rather than classical, since intuitionistic logic admits a larger class of rules that have "definite" properties. Since our basic language is intuitionistic, though, another novel aspect of our work is that we then write the circumscription axiom as a sentence in *second-order intuitionistic logic.* Although we state some of the properties of intuitionistic circumscription in Section 4, a more detailed discussion will be presented in a forthcoming paper [McCarty, 1991].

2 Definite Rules

Horn clauses provide the standard examples of definite rules, for a simple reason. Every set of positive Horn clauses has a unique minimal Herbrand interpretation [van Emden and Kowalski, 1976; Apt and van Emden, 1982], or, equivalently, an *initial model* [Goguen and Meseguer, 1986; Makowsky, 1985]. As a result, a set of positive Horn clauses, R, has both the *disjunctive property* and the *existential property:* A disjunction of atomic formulae, $A \vee B$, is entailed by R if and only if $\models$ or R $\models B$, and an existentially quantified atomic formula, $(\exists \mathbf{x}) A(\mathbf{x})$, is entailed by R if and only if $\mathcal{R} \models A(\mathbf{x})\theta$ for some ground substitution θ. Closely related is a proof-theoretic property, the existence of *linear proofs.* Indeed, it is fair to say that it is the *procedural interpretation* of a declarative semantics, embodied in SLD-resolution [Lloyd, 1987], that makes Horn-clause logic such a powerful tool for logic programming, for deductive databases and for knowledge representation in general.

It is well known that these useful properties are lost if we move beyond the Horn-clause subset of first-order (classical) logic. However, we can preserve these properties for a larger class of rules if we use first-order *intuitionistic logic.* Consider the following example:

$$\begin{aligned} \mathbf{SterileContainer}(x) \Leftarrow & \\ & \mathbf{Container}(x) \wedge \\ & (\forall y)[\mathbf{Dead}(y) \Leftarrow \mathbf{Bug}(y) \wedge \mathbf{Inside}(y, x)] \end{aligned} \tag{1}$$

This expression should be read: "If every bug *y* inside container *x* is a dead bug, then x is a sterile container/' Recall that a positive Horn clause is an implication with an atomic conclusion and with an antecedent consisting of a conjunction of atomic formulae. But rule (1) has a Horn clause "embedded" in its antecedent. We call such rules *embedded implications* [McCarty, 1988a; McCarty, 1988b]. Intuitionistic embedded implications have been studied by several researchers [Gabbay and Reyle, 1984; Gabbay, 1985; McCarty, 1988a; McCarty, 1988b; Miller, 1989: Hallnas and Schroeder-Heister, 1988; Bonner *ct* a/., 1989J, and have been shown to be useful for hypothetical reasoning [Bonner, 1988], for legal reasoning [McCarty, 1989], for modular logic programming [Miller, 1989]. and for natural language understanding [Pareschi, 1988].

Note that a set of rules in the form (1), interpreted classically, would give us full first-order logic. However, interpreted intuitionistically, these rules give us a proper subset of first-order lock with interesting semantic properties [McCarty, 1988a].[1] First, a set of intuitionistic embedded implications R has a *final Kripke model,* which we denote by K*. This means: Given any Kripke model K of R, there exists a unique homomorphism τ from K into K*. Second, K* is *generic* for Horn-clause queries. This means: A Horn clause query ψ is entailed by R if and only if is true in K*. Third, K* has a *unique minimal substatc,* denoted by $\sqcap \mathbf{K}^*$. Taken together, these results imply that R has the disjunctive and existential properties for Horn clause queries, a straightforward generalization of the fact that a set of (classical) positive Horn clauses has the disjunctive and existential properties for atomic queries. For this reason, it is appropriate to refer to rules in the form (1) as "definite" rules.

The proof theory for intuitionistic embedded implications is also a straightforward generalization of the proof theory for classical Horn-clause logic [McCarty, 1988b]. Consider a rulebase consisting of (1) plus the following two rules:

$$\mathbf{Dead}(y) \Leftarrow \mathbf{Bug}(y) \wedge \mathbf{Heated}(y) \tag{2}$$

$$\begin{aligned} \mathbf{Heated}(y) \Leftarrow & \ \mathbf{Container}(x) \wedge \mathbf{Heated}(x) \wedge \\ & \mathbf{Inside}(y, x) \end{aligned} \tag{3}$$

Assume a database of assertions: 'Container(p)' and *Heated(p)', and consider the query: '$(\exists x)$SterileContainer(x)?'. Our proof procedure begins by constructing an SLD-refutation tree in an *initial tableau, T*o, and proceeds until it has reduced the original query to the goal: $(\forall y)[\mathbf{Dead}(y) \Leftarrow \mathbf{Bug}(y) \wedge \mathbf{Inside}(y, \mathbf{p})]$. Note that this goal is a universally quantified implication. At this point, the proof procedure: (i) replaces the variable *y* with a special constant '!y_1' and (ii) constructs an *auxiliary tableau, T_1,* with a goal 'Dead(!yi)' and a database consisting of the assertions 'Bug(!y_1)' and 'Inside(!yi,p)\ This goal succeeds, using rules (2) and (3), which means that the universally quantified implication also succeeds. Thus the overall proof succeeds, with a definite answer substitution σ: $\{\mathbf{p} \leftarrow x\}$. For additional examples of such proofs, see [McCarty, 1988b].

3 Indefinite Information

For common sense reasoning, definite rules by themselves appear to be insufficient. Consider the following example, discussed by Moore [1982]. There are three blocks: Block(a), Block(b), Block(c); and two colors: Red and Green. The blocks are stacked in the following way: On(a,b), On(b,c); and the top and bottom blocks are

[1] The standard semantics for intuitionistic logic is Kripke semantics. See [Kripke, 1965; Fitting, 1969].

painted green and red, respectively: Green(a), Red(c). We do not know the color of the middle block, but we know that every block is painted either red or green:

$$\mathbf{Block}(x) \Rightarrow \mathbf{Red}(x) \vee \mathbf{Green}(x) \quad (4)$$

We want to know if there is something green on something red, a situation that could be represented by the following predicate:

$$\mathbf{GreenOnRed} \Leftarrow \mathbf{On}(x, y) \wedge \mathbf{Green}(x) \wedge \mathbf{Red}(y) \quad (5)$$

Intuitively, the answer should be: Yes. Either 'b' is a red block, in which case 'a' and 'b' constitute a green-on-red pair, or else 'b' is a green block, in which case 'b' and 'c' constitute a green-on-red pair.

How should we represent indefinite information of this sort? One approach is to add two new types of rules to our language:

$$P(\mathbf{x}) \Rightarrow Q_1(\mathbf{x}) \vee Q_2(\mathbf{x}) \quad (6)$$

$$P(\mathbf{x}) \Rightarrow (\exists \mathbf{y}) Q(\mathbf{x}; \mathbf{y}) \quad (7)$$

We call these rules *disjunctive* and *existential assertions,* respectively. The addition of disjunctive and existential assertions to a set of embedded implications is equivalent to full first-order intuitionistic logic, of course, but there are reasons to write a rulebase in this special form. Although we no longer have the properties of definite rules discussed in Section 2, it makes sense to stay as close to these properties as possible. We thus show in [McCarty, 1990] that a set of rules in the form (1), (6) and (7) has a final Kripke model, I*, although I* does not, in general, have a unique minimal substate. We also investigate in [McCarty, 1990] an extension of the linear proof procedure in [McCarty, 1988b] which uses a limited "disjunctive splittiig" operation whenever it encounters a rule in the form (6), and we show that this proof procedure is complete for intuitionistic (but not classical) logic. These results are related to recent work on *disjunctive logic programming* [Minker and Rajasekar, 1990; Loveland, 1987; Loveland, 1988].

But do rules in the form (6) and (7) really capture our common sense reasoning with indefinite information? Suppose the following definition is part of the 'blocks world':

$$\mathbf{Above}(x, y) \Leftarrow \mathbf{On}(x, y) \quad (8)$$

$$\mathbf{Above}(x, y) \Leftarrow \mathbf{On}(x, z) \wedge \mathbf{Above}(z, y) \quad (9)$$

and suppose we have been told that 'Above(a,b)' is true according to this definition. Is there something on 'b'? Intuitively, the answer should be: Yes. But it is standard practice [Kowalski, 1979] to write the 'only if' half of definition (8)~(9) as follows:

Above(x,y) => On(x, *y)* V (10)
(3z)[Qn(x, z) A Above(z, *y)]*

And using (10), the formal answer to our query is: No. It is straightforward to verify that the final Kripke model for rules (8)-(10) includes a substate that contains an infinite sequence of 'Above' relations, and no relation in the form 'On(w,b)\ For example:

$$\begin{array}{ll} & \mathbf{Above}(a, b), \\ \mathbf{On}(a, c_1), & \mathbf{Above}(c_1, b), \\ \mathbf{On}(c_1, c_2), & \mathbf{Above}(c_2, b), \\ \mathbf{On}(c_2, c_3), & \mathbf{Above}(c_3, b), \\ \vdots & \vdots \end{array}$$

Moreover, there is *no* set of first-order rules that gives us the intuitively correct result in this case, since transitive closure cannot be defined in first-order logic [Aho and Ullman, 1979]. Is there an alternative?

4 Circumscription

In this section, we consider a radically different approach to the representation of indefinite information. Assume that there exists a set of *definite* rules that can be applied to a world of *definite* facts. Assume also that someone else has observed the world, applied the rules, and reported some of these definite conclusions. Our job is to make inferences about the actual state of the world, even though we have not observed it directly. For example, we might be told that block 'a' is above block 'b\ using the definition of 'Above' in rules (8)-(9), and we might want to know whether there is something on 'b' The world could be in infinitely many different states, with infinitely many different configurations of 'On' facts, all supporting the conclusion 'Above(a,b)' Our information about the world is thus highly *indefinite.* Intuitively, however, we ought to be able to conclude that '$(\exists w)\mathbf{On}(w,\mathbf{b})$' is true.

The formal machinery we need for this approach is provided by McCarthy's theory of *circumscription* [McCarthy, 1980; McCarthy, 1986]. Since we are working with intuitionistic logic, however, we need to use an *intuitionistic* version of the circumscription axiom. Let R be a finite set of embedded implications, and let $P = < P_1, P_2, \ldots, P_k >$ be a tuple consisting of the "defined predicates" that appear on the left-hand sides of the rules in R. Let R*(P)* denote the conjunction of the rules in R, with the predicate symbols in *P* treated as free parameters, and let R*(X)* be the same as R*(P)* but with the predicate constants $< P_1, P_2, \ldots, P_k >$ replaced by predicate variables $< X_1, X_2, \ldots, X_k >$.

Definition 4.1: The *circumscription axiom* is the following sentence in second order intuitionistic logic:[2]

$$\mathcal{R}(P) \wedge (\forall X)[\mathcal{R}(X) \wedge \bigwedge_{i=1}^{k} (\forall \mathbf{x})[X_i(\mathbf{x}) \Rightarrow P_i(\mathbf{x})] \Rightarrow \bigwedge_{i=1}^{k} (\forall \mathbf{x})[P_i(\mathbf{x}) \Rightarrow X_i(\mathbf{x})]]$$

We denote this expression by Ctrcum(R(P);P), and we refer to it as "the circumscription of *P* in R(P)."

This axiom has the same intuitive meaning that it has in classical logic: It states that the extensions of the predicates in *P* are as small as possible, given the constraint

[2] We define second-order intuitionistic logic precisely in [McCarty, 1990], following standard accounts such as [Troelstra and van Dalen, 1988].

that R*(P)* must be true. Since the logic is intuitionistic, however, the axiom minimizes extensions at *every* substate of *every* Kripke model that satisfies R.[3]

If R consists of rules (8)-(9), then the circumscription of'Above' in R forces 'Above' to be the transitive closure of'On' [McCarthy, 1980; Lifschitz, 1985], and this entails the following implication:

$$(\exists w)\mathbf{On}(w, \mathbf{b}) \Leftarrow \mathbf{Above}(\mathbf{a}, \mathbf{b}) \quad (11)$$

We thus have a solution to the problem posed at the beginning of this section.

How might we compute such inferences, in general? We discuss this question in [McCarty, 1990], and we summarize our results briefly here. Let us formulate the general query problem as follows:

$$\mathcal{Q} \cup \mathit{Circum}(\mathcal{R}(\mathcal{P});\mathcal{P}) \models \psi \quad (12)$$

where *Q* and R are embedded implications, and ψ is an implication in the form (11) with a positive disjunctive or existential conclusion and an antecedent consisting of a conjunction of atomic formulae. Our approach is to construct a *final Kripke model* for $\mathcal{Q} \cup \mathit{Circum}(\mathcal{R}(\mathcal{P});\mathcal{P})$ under various assumptions about *Q* and R, and then to show that this final Kripke model is *generic* for the query ψ. In the present paper, we will only consider the case in which R is a set of Horn clauses, but we will analyze the general case of embedded implications in a forthcoming paper [McCarty, 199l]. For Horn clauses, the construction of the final Kripke model is simple, and is related to recent results of Kolaitis and Papamitriou [1990]. First, let K* be the final Kripke model for R itself, as defined in Section 2. Now let *B* be the set of *base predicates* in R, that is, the predicates that do not appear in *P,* and let b be the Herbrand base constructed using *B* alone. Define:

$$\mathbf{C}^* = \{\sqcap \mathbf{K}^*(s) \mid s \in 2^{\mathbf{b}}\}$$

where K*(s) denotes the set of substates *s'* in K* such that $s' \geq s$. Intuitively, C* consists of the set of least fixpoints of the Horn clauses in *ft* applied to all possible combinations of ground atomic formulae that are constructible using the predicates in *B.* We then have the following result:

Theorem 4.2: Let R be a set of Horn clauses, and let *Q* be a set of embedded implications. Let C be the largest subset of C* that satisfies *Q.* Then C is a final Kripke model for $\mathcal{Q} \cup$ *Circum(7t(P);P).*

Since we can also show that final Kripke models are generic for queries in the form ψ, we can solve the query problem (12) by showing that ψ is true in C.

In the remainder of this paper, we will investigate two ways to do this using certain additional assumptions about the form of Q and R. As an illustration of our techniques, we will work with a single example that combines the two examples in Section 3. Let R be the following set of rules:

$$\text{ChristmasBlock}(x) \Leftarrow \mathbf{Block}(x) \wedge \mathbf{Red}(x) \quad (13)$$

$$\mathbf{ChristmasBlock}(x) \Leftarrow \mathbf{Block}(x) \wedge \mathbf{Green}(x) \quad (14)$$

$$\mathbf{OnCB}(x, y) \Leftarrow \mathbf{ChristmasBlock}(x) \wedge \mathbf{ChristmasBlock}(y) \wedge \mathbf{On}(x, y) \quad (15)$$

$$\mathbf{AboveCB}(x, y) \Leftarrow \mathbf{OnCB}(x, y) \quad (16)$$

$$\mathbf{AboveCB}(x, y) \Leftarrow \mathbf{OnCB}(x, z) \wedge \mathbf{AboveCB}(z, y) \quad (17)$$

Intuitively, rules (13)-(14) define the concept of a 'ChristmasBlock', and rules (15)-(17) define the concept of a stack of 'ChristmasBlocks'. Suppose we are told that there exists a stack of 'ChristmasBlocks' in which block 'a' is above block 'b' and furthermore that 'a' and 'b' are painted green and red, respectively. Does it follow that there is something green on something red? Intuitively, the answer should be: Yes. Formally, we can pose this question by circumscribing the predicates 'ChristmasBlock,' 'OnCB' and 'AboveCB' in rules (13)-(17), adding rule (5) to *Q,* and then asking whether the following implication is entailed:

$$\mathbf{GreenOnRed} \Leftarrow \mathbf{AboveCB}(\mathbf{a}, \mathbf{b}) \wedge \mathbf{Green}(\mathbf{a}) \wedge \mathbf{Red}(\mathbf{b}) \quad (18)$$

We will show how to solve this problem in the following two sections of the paper, using two different methods.

5 Prototypical Proofs

In this section, we consider a class of inductive proofs in which the induction schema takes the form of an intuitionistic embedded implication. We can think of these proofs as having two parts: The first part is a *prototypical proof,* and it is guaranteed to exist whenever the query ψ is entailed by $\mathcal{Q} \cup$ *Circum(R(P);P).* The second part involves the proof of an embedded implication with an *embedded second-order universal quantifier,* and it is conjectured to exist whenever the prototypical proof succeeds. Although second-order intuitionistic logic is incomplete, in general, the fragment of second-order logic that we use to state the induction schema happens to have a complete proof procedure. This means that it is possible to automate the search for a solution to our sample problem, and to certain other similar problems.

First, note that rules (13)-(15) in our sample problem are nonrecursive Horn clauses. For such rules, the solution is the same in intuitionistic logic as it is in classical logic [Reiter, 1982; Lifschitz, 1985]. Let *Comp(7t)* denote Clark's Predicate Completion [Clark, 1978]. We then have the following result, which is proven in [McCarty, 1990]:

Theorem 5.1: Let R be a set of nonrecursive Horn clauses. Then *Circum(R(P);P)* is equivalent to *Comp(R).*

The remaining rules in our sample problem have a simple form, suggesting the following:

Definition 5.2: R is a *linear recursive definition* of the predicate *A* if it consists of:

1. A Horn clause with 'A(X)' on the left-hand side and a conjunction of nonrecursive predicates on the right-hand side, and
2. A Horn clause that is linear recursive in *A,*

[3]Note that we have written the circumscription axiom as a second-order universally-quantified embedded implication. Alternative versions, using negation and second-order existential quantification, which are equivalent in classical logic, would not be equivalent intuitionistically.

Let '$A(\mathbf{x})\Rightarrow A^0(\mathbf{x})$' be the rule obtained from (1) by applying Clark's Predicate Completion. We say that '$A(\mathbf{x})\Rightarrow A^0(\mathbf{x})$' is the *prototypical* definition of $A(\mathbf{x})$.

Let '$X(\mathbf{x})\Rightarrow\Delta X(\mathbf{x})$' be the rule obtained from (2) by applying Clark's Predicate Completion and then replacing the predicate constant A with the predicate variable X. We say that '$X(\mathbf{x})\Rightarrow\Delta X(\mathbf{x})$' is the *transformation* associated with $A(\mathbf{x})$. □

In particular, rules (16)–(17) constitute a linear recursive definition of the predicate 'AboveCB', in which

$$\mathrm{AboveCB}(x,y) \Rightarrow \mathrm{OnCB}(x,y) \tag{19}$$

is the prototypical definition, and

$$X(x,y) \Rightarrow (\exists z)[\mathrm{OnCB}(x,z) \wedge X(z,y)] \tag{20}$$

is the transformation. Although (20) is written as an implication, it should be viewed, quite literally, as an operation that transforms any relation between x and y that matches its left-hand side into a relation between x and y that matches its right-hand side. Comparing the components of Definition 5.2 with (19) and (20), we now extract '$\mathrm{AboveCB}^0(x,y)$' and '$\Delta X(x,y)$' from (19) and (20), and we use them to define the induction schema that we need to solve our problem.

Let $\Phi(A)$ be any Horn clause in which the predicate constant A appears on the right-hand side. For example:

$$\Phi(A) \equiv (\forall \mathbf{x})\left[P(\mathbf{x}) \Leftarrow A(\mathbf{x}) \wedge \bigwedge_{i=1}^{l} B_i(\mathbf{x})\right]$$

We will treat $\Phi(A)$ as a schema that depends on A, so that we are free to substitute A^0, ΔX and X as we wish. Thinking about (20) as a transformation suggests:

Definition 5.3: The *induction schema* for $\Phi(A)$ is the following sentence in second-order intuitionistic logic:

$$\Phi(A) \Leftarrow \Phi(A^0) \wedge (\forall X)[\Phi(\Delta X) \Leftarrow \Phi(X)]$$

The interesting point about this induction schema is that it takes the form of an embedded implication with an embedded second-order universal quantifier. Second-order intuitionistic logic has no complete proof procedure, of course, but it turns out that a set of second-order rules in this form *does* have a complete proof procedure. The proof procedure is actually very simple, and it can be understood by analogy to the first-order intuitionistic proof procedure described in Section 2. Recall how we handled a goal with a first-order universal quantifier, $(\forall y)$. We simply created a special constant '$!y_1$' and showed that the implication succeeded when y was replaced by '$!y_1$'. Similarly, when we encounter a goal with a second-order universal quantifier, $(\forall X)$, we simply create a special predicate constant '$!\mathbf{X}_1$' and we try to show that the implication succeeds when X is replaced by '$!\mathbf{X}_1$'. For a proof that this procedure is complete, see [McCarty, 1990].

Let us now see how the induction schema in Definition 5.3 can be applied to solve our sample problem. Since we are trying to prove the implication in (18), we construct an initial tableau, T_0, with 'AboveCB(a,b)', 'Green(a)' and 'Red(b)' in its data base, and with 'GreenOnRed' as its goal. Our first step is to show, if possible, that this proof succeeds using the prototypical definition in (19). However, the reader can easily verify that there exists an SLD-refutation proof of 'GreenOnRed' in this tableau using rules (5) and (19), and using *Comp(R)* applied to rule (15). Moreover, from an inspection of this proof, it is apparent that there also exists a proof of the following universally quantified implication:

$$(\forall x)[\mathrm{GreenOnRed} \Leftarrow \mathrm{OnCB}(x,\mathrm{b}) \wedge \mathrm{Green}(x) \wedge \mathrm{Red}(\mathrm{b})] \tag{21}$$

Let us call this implication $\Phi(\mathrm{AboveCB}^0)$. Then $\Phi(\mathrm{AboveCB})$ is the following universally quantified implication:

$$(\forall x)[\mathrm{GreenOnRed} \Leftarrow \mathrm{AboveCB}(x,\mathrm{b}) \wedge \mathrm{Green}(x) \wedge \mathrm{Red}(\mathrm{b})] \tag{22}$$

and if we can prove (22) we will also have a proof of our original query (18). Therefore, using the induction schema in Definition 5.3, we try to prove $(\forall X)[\Phi(\Delta X) \Leftarrow \Phi(X)]$. This goal is an implication with a second-order universal quantifier, so we create a new tableau, T_1, we add $\Phi(!\mathbf{X}_1)$ to the data base, and we try to prove $\Phi(\Delta!\mathbf{X}_1)$ in T_1.

Let us write out each of these schemata in detail, $\Phi(!\mathbf{X}_1)$ is the following implication:

$$(\forall x)[\mathrm{GreenOnRed} \Leftarrow !\mathbf{X}_1(x,\mathrm{b}) \wedge \mathrm{Green}(x) \wedge \mathrm{Red}(\mathrm{b})] \tag{23}$$

and $\Phi(\Delta!\mathbf{X}_1)$ is equivalent to the following implication:

$$(\forall x,z)[\mathrm{GreenOnRed} \Leftarrow \mathrm{OnCB}(x,z) \wedge !\mathbf{X}_1(z,\mathrm{b}) \wedge \mathrm{Green}(x) \wedge \mathrm{Red}(\mathrm{b})] \tag{24}$$

To prove (24), we instantiate x and z to the special constants '$!\mathbf{x}_1$' and '$!\mathbf{z}_1$', we add the right-hand side of (24) to the data base of T_1, and we try to prove the left-hand side of (24). The remaining details are somewhat tedious, but the reader should be able to check that this proof does in fact go through. The main point to note is that the proof now uses *Comp(R)* applied to rules (13) and (14), which yields a disjunctive assertion. We thus need to apply the "disjunctive splitting" operation discussed in Section 3. When this final step succeeds, however, the inductive proof itself succeeds, and the query is shown to be true.

The justification for our approach can be found in the following two theorems, which are proven in [McCarty, 1990] using our results on final Kripke models. In the statement of these theorems, *S(A)* denotes the set of all induction schemata for *A* that can be constructed using Definition 5.3, and *V(A)* denotes the prototypical definition of *A* given by Definition 5.2. Both theorems apply to the situation in which R is a linear recursive definition, *Q* is a set of embedded implications, and ψ is an implication with a positive disjunctive or existential conclusion and an antecedent consisting of a conjunction of atomic formulae.

Theorem 5.4:

$$\mathcal{Q} \cup \mathcal{R} \cup \mathcal{S}(A) \models \psi \Longrightarrow \mathcal{Q} \cup \mathit{Circum}(\mathcal{R}(A); A) \models \psi$$

Theorem 5.5:

$$\mathcal{Q} \cup \mathcal{R} \cup \mathcal{P}(A) \models \psi \Longleftarrow \mathcal{Q} \cup \mathit{Circum}(\mathcal{R}(A); A) \models \psi$$

Theorem 5.4 tells us that inductive proofs are sound but not necessarily complete, while Theorem 5.5 tells us that prototypical proofs are complete but not necessarily sound. Together, these theorems sanction the strategy we illustrated in our example: Try to find a prototypical proof first, and then use this proof to suggest a suitable induction schema.

Our sample problem is still relatively simple, but we have constructed proofs of this sort for more difficult problems. In particular, we have applied our techniques to prove various properties of PROLOG programs [Kanamori and Fujita, 1986; Elkan and McAllester, 1988]. For example, let 'Append(l,m,n)' be defined as usual. Let 'Reverse(r,*)' be defined as follows:

Reverse(nil, nil)

$$\mathbf{Reverse}([q \mid r], p) \Leftarrow \mathbf{Reverse}(r, s) \wedge \mathbf{Append}(s, [q], p)$$

We can then show that '$(\forall x)(\forall y)$ [Reverse*(x,y)* $\Leftarrow$ Rever8e(y,x)]' is entailed by the circumscription of 'Append' and 'Reverse' in this rulebase. For the details of this proof, see [McCarty, 1990].

6 A Decision Procedure

We have shown in Theorem 5.4 that a certain class of induction schemata provides a sound inference procedure for circumscribed definite rules. Furthermore, the structure of these schemata allows us to use a complete proof theory for second-order embedded implications in the inductive step of the proof. This raises the question: Is the resulting proof theory for the circumscribed rulebase itself complete? We show in this section that it is not. In fact there can be no such proof theory, since the query problem will be shown to be not even semi-decidable. Nevertheless, we will demonstrate that one can still find interesting classes of decidable queries. Our results are significant since they show that, even in cases where the circumscription of a theory is not first-order equivalent, it is possible to decide certain broad classes of queries. We refer the reader to [van der Meyden, 1990] for proofs of the results in this section.

For the remainder of this section we restrict our attention to rules R which contain only positive Horn clauses without function symbols, i.e., all programs are DATALOG programs [Chandra and Harel, 1982]. Furthermore, we require that there be no repeated variables in the heads of rules.[4] We consider the following restricted formulation of the query problem:

$$D \cup \mathit{Circum}(\mathcal{R}(P); P) \models \phi \qquad (25)$$

where D is a set of ground base and defined atoms, and ϕ is a closed positive existential formula in the base and defined predicates, i.e., a formula constructed using only the operators $\wedge, \vee$ and $\exists$. The proof of the following result is by an encoding of the containment problem for context-free languages:

[4] This last restriction is made to simplify the presentation only; our results still hold if it is removed.

Theorem 6.1; For arbitrary DATALOG rules R, sets of ground atoms D and positive existential queries ϕ, the problem $D \cup \mathit{Circum}(7l(P);P) \models \phi$ is undecidable.

We will now show that the complement of the query problem is recursively enumerable. Define an *expansion* of a defined atom *A(x)* by R to be any set *E(x)* of base atoms such that either

1. $E = \{B_1(x;b), \ldots, B_n(x;b)\}$ for some tuple b of *new* constants, and for some rule

$$A(\mathbf{x}) \Leftarrow \bigwedge_{i=1}^{n} B_i(\mathbf{x};\mathbf{y})$$

in R such that all the B_i are base predicates, or

2. $E = \{B_1(\mathbf{x};\mathbf{b}), \ldots, B_n(\mathbf{x};\mathbf{b})\} \cup \bigcup_{j=1}^{m} E_j(\mathbf{x};\mathbf{b})$ for some tuple b of *new* constants, and for some rule

$$A(\mathbf{x}) \Leftarrow \bigwedge_{i=1}^{n} B_i(\mathbf{x};\mathbf{y}) \wedge \bigwedge_{j=1}^{m} A_j(\mathbf{x};\mathbf{y})$$

in R such that all the B_i are base predicates, all the A_j are defined predicates, and each $E_j(x;y)$ is an expansion of *Aj(x;* y) by R.

Also, if a is a tuple of constants, then we say that E(a) is an expansion of A (a) by R if *E(x)* is an expansion of *A(x)* by R. We write $\mathit{Expand}_R(A)$ for the set of expansions of A by R.. If D is a set of ground atoms in both base and defined predicates, then an expansion of D by R is any set of ground atoms obtained from D by replacing each defined atom $A \in D$ by an expansion of A by R.

For example, let R consist of the rules (13)-(17). Then the set:

- **{Block(a), Green(a), On(a, b), Block(b), Red(b)}**

is an expansion of 'AboveCB(a, b)', and the set:

- **{Green(a), Block(a), Red(a), On(a, c), Block(c), Red(c), On(c, b), Block(b), Red(b)}**

is an expansion of D = {Green(a), AboveCB(a, b), **Red(b)}.**

The following proposition shows that we may restrict attention to a denumerable set of models of a particular form when answering queries:

Lemma 6.2: $D \cup \mathit{Circum}(\mathcal{R}(P);P) \models \phi$ **if and only if** $M \models \phi$ **for all expansions** M **of** D **by** $\mathcal{R}$.

Lemma 6.2 shows that the problem of deciding that a query is *not* entailed is semi-decidable, since it suffices to find a single expansion of D in which the query fails. It follows from this and Theorem 6.1 that $\{(D, \mathcal{R}, \phi) | D \cup \mathit{Circum}(\mathcal{R}(P);P) \models \phi\}$ is not a recursively enumerable set. Thus the techniques of Section 5 can only provide sufficient conditions for answering queries.

In spite of these undecidability results, there exist broad classes of queries for which the query problem can be shown to be decidable, even in cases when the circumscription of R is not equivalent to a first-order theory. Define a query to be *basic* if it contains occurrences of base predicates only.

Theorem 6.3: For arbitrary DATALOG rules $\mathcal{R}$, sets of ground atoms D and basic queries ϕ, the problem $D \cup Circum(\mathcal{R}(P);P) \models \phi$ is decidable.

The decision procedure involves a sort of "finite model property". If ϕ is a basic query, let the equivalence relation $\equiv_\phi$ be defined on the expansions of a defined atom $A(\mathbf{x})$ as follows: $E_1(\mathbf{x}) \equiv_\phi E_2(\mathbf{x})$ when for all sets D of ground base atoms and for all constants $\mathbf{a}$, $D \cup E_1(\mathbf{a}) \models \phi$ if and only if $D \cup E_2(\mathbf{a}) \models \phi$. Intuitively, this relation holds just when the expansion E_1 makes the same "contribution" to the satisfaction of ϕ as the expansion E_2. For basic queries ϕ, it can be shown that the relation $\equiv_\phi$ has a finite number of equivalence classes and that $E_1(\mathbf{x}) \equiv_\phi E_2(\mathbf{x})$ is decidable. Let $Rep_\phi(\mathcal{R}, A)$ be a set of *representatives* of the equivalence classes of the expansions in $Expand(\mathcal{R}, A)$, i.e., for each expansion $E \in Expand(\mathcal{R}, A)$ there exists $E' \in Rep_\phi(\mathcal{R}, A)$ such that $E \equiv_\phi E'$. Then one can show the following:

Lemma 6.4: $D \cup Circum(\mathcal{R}(P);P) \models \phi$ if and only if $M \models \phi$ for all expansions M of D constructed by replacing each defined atom $A \in D$ by some expansion from $Rep_\phi(\mathcal{R}, A)$.

A (naive) decision procedure for answering basic queries thus works as follows:

1. For each defined predicate A, compute $Rep_\phi(\mathcal{R}, A)$.
2. Verify that all expansions of D using these representatives satisfy ϕ.

This simple procedure does not have optimal complexity, however, and we refer the reader to [van der Meyden, 1990] for a detailed analysis of the complexity of this problem.

Returning to our "stack of Christmas blocks" example, let $\mathcal{R}$ consist of the rules (13)-(17), and let ϕ be:

$$(\exists x)(\exists y)[\mathrm{On}(x, y) \wedge \mathrm{Green}(x) \wedge \mathrm{Red}(y)]$$

It can be shown that $Rep_\phi(\mathcal{R}, \mathrm{AboveCB}(x, y))$ is a subset of the set of expansions of "depth" less than or equal to two. That is, all these expansions are of the form:

1. $\{\mathrm{Block}(x), C_1(x), \mathrm{On}(x, y), \mathrm{Block}(y), C_2(y)\}$ or
2. $\{\mathrm{Block}(x), C_1(x), \mathrm{On}(x, z), \mathrm{Block}(z), C_2(z), C_3(z), \mathrm{On}(z, y), \mathrm{Block}(y), C_4(y)\}$

where the predicates C_i are either 'Red' or 'Green'.

In [van der Meyden, 1990], it is shown that there are additional classes of decidable queries:

Theorem 6.5: Let $\mathcal{R}$ be a rulebase that includes only *monadic* defined predicates P (but which may include base predicates of arbitrary arity). Then $D \cup Circum(\mathcal{R}(P);P) \models \phi$ is decidable for sets of atoms D and queries ϕ containing the defined predicates P as well as the base predicates.

Define a rulebase $\mathcal{R}$ with just one defined predicate P to be *singular* if there exists an index i such that each rule $P(x_1 \ldots x_n) \Leftarrow B$ has all occurrences of P in the body B of the form $P(x_1 \ldots x_{i-1}, y, x_{i+1} \ldots x_n)$ for some variable y. For example, the rules (8) and (9) defining the relation 'Above' constitute a singular rulebase.

Theorem 6.6: If $\mathcal{R}$ is a singular rulebase defining the predicate P, then $D \cup Circum(\mathcal{R}(P); P) \models \phi$ is decidable for sets of atoms D and queries ϕ containing the predicate P as well as the base predicates.

These results are all consequences of a theorem of Courcelle [1990] concerning graph grammars. In general, we retain decidability when we permit in the query any predicate to which the rulebase gives a definition expressible in monadic second-order logic. Unfortunately, it can be shown to be undecidable whether a rulebase defines a predicate expressible in monadic second-order logic, so we must be content with enumerating special cases, as above, if we wish to go beyond the class of basic queries.

7 Conclusion

We have presented two approaches to answering queries in the presence of indefinite information, both of which are able to handle the "stack of Christmas blocks" example. The reader may have wondered why, since the method of Section 6 is a decision procedure, one would bother with the prototypical proofs of Section 5? The reason is that the the decision procedure works for a smaller class of rules than the prototypical proofs. On the other hand, the price paid by the prototypical proofs for their ability to deal with a potentially larger set of examples is logical incompleteness. It would be interesting to determine the extent to which this incompleteness corresponds to the incompleteness of human reasoning when faced with indefinite information of comparable logical complexity.

References

[McCarty, 1988a] L.T. McCarty. Clausal intuitionistic logic. I. Fixed-point semantics. *Journal of Logic Programming,* 5(1):1-31, 1988.

[McCarty, 1988b] L.T. McCarty. Clausal intuitionistic logic. II. Tableau proof procedures. *Journal of Logic Programming,* 5(2):93-132, 1988.

[McCarty, 1990] L.T. McCarty. Computing with prototypes. Technical Report LRP-TR-22, Computer Science Department, Rutgers University, 1990. A preliminary version of this paper was presented at the *Bar Ran Symposium on the Foundations of Artificial Intelligence,* Ramat Gan, Israel, June 1989.

[McCarty, 1991] L.T. McCarty. Circumscribing embedded implications. In A. Nerode et al., editors, *Proceedings, First International Workshop on Logic Programming and Non-Monotonic Reasoning,* page (forthcoming). MIT Press, 1991.

[van der Meyden, 1990] R. van der Meyden. Recursively indefinite databases. In S. Abiteboul and P.C. Kanellakis, editors, *Proceedings of the Third International Conference on Database Theory,* pages 364-378. Springer LNCS No. 470, 1990.

Note: These references are truncated because of space limitations. A full list of references is available from the authors on request.