CMU-CS-84-154

Circumscription and Implicit Definability

Jon Doyle

Computer Science Department
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213
U.S.A.

Abstract: We explore some connections between the technique of circumscription in artificial intelligence and the notion of implicit definition in mathematical logic. Implicit definition can be taken as the informal intent, but not necessarily the formal result, of circumscription. This raises some questions for logical theory and suggests some implications for artificial intelligence practice. The principal implication is that when circumscription "works" its conclusions can be explicitly described.

August 31, 1984

This paper will be presented at the AAAI Workshop on Non-Monotonic Reasoning.

This research was supported by the Defense Advanced Research Projects Agency (DOD), ARPA Order No. 3597, monitored by the Air Force Avionics Laboratory under Contract F33615-81-K-1539. The views and conclusions contained in this document are those of the author, and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Government of the United States of America.

Introduction

John McCarthy [1977, 1980] introduced the notion of circumscription into artificial intelligence as a means for handling his "qualification problem" by jumping to conclusions. The detailed statement is presented below, but briefly put, the idea of circumscription is to rule out all objects and conditions not explicitly entailed by one's beliefs, as a way of avoiding explicit consideration of the myriad potential exceptional cases consistent with one's incomplete knowledge.

Richard Statman pointed out to me that an old idea of mathematical logic, ***implicit definability***, seems essentially the same as circumscription, and that much is known in mathematical logic about implicit definability. The following examines some connections between these ideas with two audiences in mind. I first briefly describe the constellation of difficulties faced in artificial intelligence that motivate circumscription, and enumerate some of its important concrete uses. I then present the connection with implicit definability, and survey some basic results from mathematical logic concerning it. Both parts attempt to convey to logicians the apparent needs of artificial intelligence and the corresponding theoretical opportunities, and to convey to artificial intelligence researchers the theoretical possibilities and relevant literature.

This paper would not exist if not for the initial observation, patient explanations, proofs and algorithms of Richard Statman. I am grateful to him for this help, and to Joseph Schatz for comments and encouragement. I, of course, am responsible for any errors in the following.

What is circumscription for?

The motivations for circumscription begin with the observation that there are too many things to think about, even in mundane decisions about what to do or what to believe. We may accept some statements about the world as our beliefs, but they have infinitely many consequences. We may see that they are incomplete, but there are infinitely many completions. And we may guess that they are wrong, but there are infinitely many changes we could make. Yet our resources, especially time, are finite, and so reasoning must be finite if we are to periodically take action and live in this world. Finiteness of reasoning and deliberation means that we must ignore almost all of these infinitely many possibilities. Considering and accepting or rejecting every possibilit; one by one will not work. Instead, we must never even consider most of them.

The problem of ignoring most possibilities is different from the problem of ignoring most impossibilities. Many techniques in artificial intelligence are directed toward the latter problem. For example, search theory provides techniques like α-β cutoffs, branch and bound, A*, B*, etc. Each of these makes use of information discovered during the search to recognize whole sectors of seeming possibilities as, in fact, impossibilities. Another example is the use of unification in resolution, where the partial information about possible ground instances of each individual clause is combined to ignore individually possible but collectively impossible substitutions. A final example is the use of dependency information in belief revision, where records of the dependence of contradictions on assumptions are used to ignore revised sets of assumptions which change only chronologically intervening but logically independent assumptions.

The problem of ignoring most possibilities is much more difficult than the problem of ignoring most impossibilities. Everyone agrees that ignoring most impossibilities is a good thing, and we have precise notions, such as logical contradictions, entailment, optimization, etc., which characterize what is impossible given some information. In contrast, ignoring possibilities requires a choice of which ones are good to ignore. We certainly do not wish to ignore all, unless we are ostriches. But what makes ignoring some possibilities more reasonable than ignoting others? Purely arbitrary cutoffs on consideration by setting fixed resource allocations are not very attractive, for different tasks and decisions call for different amounts of effort. Simply stopping after 25 or 10^{25} steps produces random results. If we wish our ignorance to be principled and deliberate, we must seek to consider or ignore possibilities based on our purposes, circumstances, and informational and computational resources, accepting blunt termination only in the final extremities of unavoidable resource exhaustion.

Artificial intelligence has pursued two complementary approaches toward achieving ignorance in reasonable ways. The first approach is to separate out restricted classes of inferences and actions which, being guaranteed to be finite, even fast, may be always performed automatically without conscious consideration or oversight, and without danger of unexpected exhaustion except in cases of unusually severe resource limitations. This does not solve the problem, but at least separates the "easy" cases of mental actions from the realm in which all actions must be deliberatively considered and confined. These definitely finite cases are also prime candidates for tailored implementations, especially parallel implementations, since the guarantees of finiteness stem from knowledge of what the computations and answers look like. Important examples of this approach include the explicit and limited sorts of inferences made by reason maintenance systems and by hierarchical inheritance systems. In these, only some of the possible inferences are made from the information possessed, namely those inferences stipulated by explicit "reasons" and "inheritance links." All other sorts of inferences are left for deliberate investigation by the larger system employing these subsystems. Though such definitely finite classes of reasoning operations may seem trivial to logicians, or when compared with the grand adventure of locating an answer in the infinite unknown, in practice the finite cases are extremely important. In each situation we must adapt our basic attitudes to the particulars of the case at hand, reworking and applying our "web of belief" or "complex of attitudes" over and over. What "common sense" and "common knowledge" are has not yet been completely understood, but it is certain that part of those features of agents is the efficient and flexible use of vast numbers of facts which, by their very commonality, enter into many of the mundane decisions we make constantly every day. Adaptations of such background knowledge handles much of the routine work, if we are lucky, leaving the agent free to concentrate its attention and resources on the non-routine problems.

The second approach toward principled ignorance taken by artificial intelligence is that of ruling out possibilities by adopting a narrow view of what is possible. For example, frequently we are forced to take action in spite of incomplete knowledge about our circumstances and the consequences of possible actions. We may wish to follow rules that indicate which actions are appropriate in which circumstances, but due to the incompleteness of our knowledge, none of these may be clearly applicable. We may have other rules in the finite realm which specify particular standard assumptions to make, but even after following these, we may still not have enough information. Unlimited exploration of the possibilities is out of the question, for there are too many. In these straits, it is common to take what we do know or have assumed as our "best guess" about the situation. We assume that what we do know is all there is, that what we do not know to be true must be false. This is the idea of both circumscription and the "closed world assumption"—each of which involves jumping to conclusions based on those delimited by our explicit knowledge.

For example, we may know of several things that lie on a table, but may need to know about all things on the table to tell if we have succeeded in removing everything from the table. In this case, our best guess is to assume that the several objects we know to be on the table are the only things there, to assume that nothing else is on the table.

One may wish to succeed in some action that has certain conditions of success, for example, rowing in a boat to the other side of a river. As McCarthy's well-known anecdotes show, we will never act unless we can rule out the myriad ways the action might fail. To do this, we assume that if something was wrong, we would know about it, and conclude that nothing is wrong from our ignorance of any problems.

Or finally, we may wish to take several actions in a row as a means to accomplishing a difficult goal, and to tell what action to do next, we need to know what circumstances result from the preceding actions. We may know some consequences of possible actions explicitly, through "laws of motion" of the form "Conditions C will obtain after taking action A in situation S." Such rules may tell us about the overt effects of actions, but it is infeasible to have such rules explictly mention the vast majority of conditions left unchanged by actions. Hence to tell what conditions result, we must guess that each action changes nothing except those things we can tell it changes from the laws of motion and our other knowledge about relations between things.

These assumptions adopted in taking the narrow view serve to convert many former possibilities into temporary impossibilities, which can then be avoided by the methods mentioned previously. But to make use of "best guesses" in artificial intelligence, we also need ways of telling exactly what our best guess should be, when it should be changed, and how to compute it from our knowledge. In particular, we must address the following questions.

(1) Precisely which conclusions make sense as "best guesses" from given information?

(2) Is there an explicit, or at least concise, characterization of these conclusions and their dependence on the given information?

(3) How can these conclusions be computed quickly? Or approximated if exact answers are uncomputable or infeasible?

(4) Is there an easy way to tell when changes to the explicit information invalidate the guesses, and when the changes leave the guesses the same?

(5) When the guesses must change, can they be quickly updated to fit the new information?

What is circumscription?

McCarthy introduced (in [1977], modified in [1980]) the formal notion of circumscription as a solution to problem (1).

The formal definition of circumscription is as follows. Fix a first order logical language $\mathcal{L}$, and let A be a set of axioms stated in $\mathcal{L}$, that is, a set of closed formulas. For convenience, when A is finite, we will sometimes confuse A with a sentence $\hat{A}$ obtained as a conjunction of the members of A.

Let P be a predicate symbol of $\mathcal{L}$ of type or arity n. We will write $P(\bar{x})$ to abbreviate $P(x_1, \ldots, x_n)$. We also write $A(Q)$ to mean the result of substituting the symbol Q for P everywhere in A, so that $A = A(P)$.

When A is finite, we define $\hat{B}(A, P)$, the *circumscription schema for P in A*, to be the sentence schema

$$[A(Q) \wedge \forall \bar{x}(Q(\bar{x}) \supset P(\bar{x}))] \;\supset\; \forall \bar{x}(P(\bar{x}) \supset Q(\bar{x})).$$

We write $B(A, P)$ to mean the set of sentences abbreviated by $\hat{B}(A, P)$, that is, the set resulting from substituting every expression of $\mathcal{L}$ of type n for Q in $\hat{B}(A, P)$. We further define $K(A, P)$, the *circumscription of P in A*, to be the union of A and $B(A, P)$, that is, $K(A, P) = A \cup B(A, P)$. When A is finite, we can write $\hat{K}(A, P)$ to mean the schema $\hat{A} \wedge \hat{B}(A, P)$.

Aside from slight differences in notation, the definition above of $\hat{B}(A, P)$ is exactly McCarthy's. McCarthy does not define the circumscription for infinite sets of axioms, and neither do we for the moment. We follow Reiter's [1982] lead in introducing notation for $K(A, P)$, though ours is different than his. McCarthy's discussion apparently abandons the original axioms A for their conditional appearance in $\hat{B}(A, P)$, focussing on the consequences of $B(A, P)$ alone. Here our terminology diverges from that of McCarthy, for we say that a sentence is a consequence of circumscribing P in A if it is among $\mathrm{Th}(K(A, P))$, while McCarthy would mean that the sentence appears among $\mathrm{Th}(B(A, P))$.

Two simple examples will be sufficient for most of our purposes. Both are adapted from [McCarthy 1980].

First, let A be the set of two axioms $\mathit{Block}(b1)$ and $\mathit{Block}(b2)$, read as stating that both $b1$ and

$b2$ are blocks. Then

$$\forall x(Block(x) \equiv [x = b1 \lor x = b2])$$

is a consequence of circumscribing *Block* in A, that is, the consequences in Th($K(A, Block)$) state that only $b1$ and $b2$ are blocks.

Second, let A contain the single axiom $Block(b1) \lor Block(b2)$. This time,

$$\forall x(Block(x) \equiv (x = b1)) \lor \forall x(Block(x) \equiv (x = b2))$$

is in Th($K(A, Block)$), stating that either $b1$ is the only block, or $b2$ is the only block.

Now that we know some of the motivations for circumscription and know its formal definition, what do we know? While circumscription clearly is relevant to the problems of artificial intelligence mentioned previously, surprisingly little seems to be known about its deeper nature, techniques of application, or even what questions to ask. McCarthy [1980], for example, presents only motivational difficulties and the formal definition. Davis [1980] discusses some existence questions, and Reiter [1982] goes further and shows that circumscription implies Kowalski's technique of "predicate completion" in simple cases of logic programming. But what does circumscription actually accomplish? It does not do exactly what its motivations lead one to expect. Consider the disjunctive block example above. Here circumscription yields not one minimal conception of what blocks are, but two. In addition to these questions, circumscription has never been studied with respect to problems (2)-(5). Artificial intelligence is lucky this time, however, for circumscription is initmately related to the logical notion of implicit definability, about which logicians know many things—not enough, perhaps, to satisfy all the practical needs of artificial intelligence, but good insights needed to better pursue those needs.

Implicit definability

Mathematical logic has long possessed the notion of ***implicit definition*** of predicates. Retaining our previous notation, a set of axioms A (finite or infinite) implicitly defines P just in case A forces a "unique" meaning for P. There are several equivalent precise statements of this intuitive idea.

First, A implicitly defines P iff when we pick some symbol Q not in $\mathcal{L}$, we can prove $\forall \vec{x}(P(\vec{x}) \equiv Q(\vec{x}))$ from $A(P) \cup A(Q)$, that is, iff $A(P), A(Q) \vdash \forall \vec{x}(P(\vec{x}) \equiv Q(\vec{x}))$. In other words, Q cannot differ from P if it satisfies the same axioms as does P.

Second, A implicitly defines P iff whenever $\mathcal{M}$ and $\mathcal{N}$ are two models of $A(P)$ ($\mathcal{M} \models A$, $\mathcal{N} \models A$) that are isomorphic for all relations besides P, then they are also isomorphic for P. In particular, if $\mathcal{N}$ is an automorphism of $\mathcal{M}$, then P is the same in both, that is, $\mathcal{N}(P) = \mathcal{M}(P)$.

Third, A implicitly defines P iff any predicate satisfying the conditions $A(P)$ on P must be P: formally, for finite A, if $A(Q) \supset \forall \vec{x}(P(\vec{x}) \equiv Q(\vec{x}))$ is valid for each expression Q of $\mathcal{L}$ of the same type as P.

If we note that

$$A(Q) \supset \forall \vec{x}(P(\vec{x}) \equiv Q(\vec{x}))$$

implies the formula

$$[A(Q) \land \forall \vec{x}(Q(\vec{x}) \supset P(\vec{x}))] \supset \forall \vec{x}(P(\vec{x}) \supset Q(\vec{x})),$$

we see that implicit definability implies that the circumscription schema $\hat{B}(A, P)$ is valid. That is, the implicit definability schema says that P is unique; the circumscription schema says that P is minimal; and of course uniqueness is a special case of minimality. Informally we might think of circumscription as producing a unique result—it does, after all, produce a single theory—that is, we might say that the intent of $K(A, P)$ is to be a theory which implicitly defines P, a theory stating both that $A(P)$ holds and that P is the unique minimal predicate forced by A.

Unfortunately, the "smallest" value for P may actually be several different minimal values for P, a possibly different one in each model of $A(P)$. In this case, $K(A, P)$ will fail to implicitly define P, so circumscription is more properly thought of as a generalization of implicit definition. In fact, few theories define predicates implicitly, even if they are constructed with the intent of doing so. While in the conjunctive block example above $K(A, \mathit{Block})$ does implicitly define *Block*, in the disjunctive example $K(A, \mathit{Block})$ does not implicitly define *Block*. There are instead two nonisomorphic models of $K(A, \mathit{Block})$, one in which $b1$ is the only block, and one in which $b2$ is the only block. These difficulties do not arise if $\forall Q[\hat{B}(A, P)]$ is taken as a second-order axiom. In this case, $A(P) \wedge \forall Q[\hat{B}(A, P)]$ does state the existence of a least model for P. Alternatively, we might strengthen the circumscription schema $\hat{B}(A, P)$ to be $A(Q) \supset \forall \vec{x}(P(\vec{x}) \equiv Q(\vec{x}))$, which would also require the circumscribed predicate to be uniquely defined. Unfortunately, these strengthenings do not result in more useful results from circimscription, for they simply yield inconsistent theories when applied to axioms like those of the disjunctive example above.

The general rarity of implicit definitions leads to many questions about circumscription. When *does* $K(A, P)$ define P implicitly? If it does not, should we care? Does not $K(A, P)$ produce useful conclusions anyway? Put another way, what does $K(A, P)$ accomplish when it fails to implicitly define P? Are there any benefits to implicit definition succeeding? And could some other additions to A succeed at implicitly defining P when circumscription fails? We treat some of these questions below.

Explicit definability

Since axiom schema are less convenient than finite axiom sets, and since the circumscription schema defines P rather indirectly, we can pursue question (2) and ask if $K(A, P)$ entails some concise characterization of P. Specifically, can we find among the consequences of $K(A, P)$ an ***explicit definition*** for P of the form $\forall \vec{x}(P(\vec{x}) \equiv \phi(\vec{x}))$, where ϕ is some formula not involving P? For example, Reiter [1982], drawing on work of Clark [1978], has shown that for the special case in which $A(P)$ is Horn in P, circumscribing P in $A(P)$ entails the predicate completion of P. That is, in this case $A(P)$ can be rewritten in the form

$$\psi \wedge \forall \vec{x}(\phi(\vec{x}) \supset P(\vec{x})),$$

and $K(A, P)$ entails the formula

$$\forall \vec{x}(P(\vec{x}) \supset \phi(\vec{x})).$$

Together these yield

$$\forall \vec{x}(P(\vec{x}) \equiv \phi(\vec{x})).$$

Our question here is whether ψ and ϕ may be chosen so that they do not mention P.

The answer to this question about the existence of explicit definitions is given by a famous result of mathematical logic. ***Beth's Definability Theorem*** states that implicit and explicit definability are equivalent: or formally, that A (finite or infinite) implicitly defines P iff there is a formula ϕ involving only the symbols of A exclusive of P such that $A \vdash \forall \vec{x}(P(\vec{x}) \equiv \phi(\vec{x}))$. This means that an explicit definition of P in $K(A, P)$ ***does not exist*** when $K(A, P)$ fails to implicitly define P. This is part of the reason why having implict definitions succeed is a good thing. Recognition of Beth's theorem changes one's outlook on circumscription, from the ill-posed problem of choosing "good" or "useful" instances of the schema to the concrete problem of computing the explicit definition of P.

But even if P admits an explicit definition, it may not be possible to eliminate the schema in favor of the explicit definition. That is, even if $K(A, P)$ is consistent and if $K(A, P) \vdash \forall \vec{x}(P(\vec{x}) \equiv \phi(\vec{x}))$ as desired, it may be that $\mathrm{Th}(K(A, P)) \not\doteq \mathrm{Th}(A \cup \{\forall \vec{x}(P(\vec{x}) \equiv \phi(\vec{x}))\})$, for the circumscription schema forces minimality of P, while an explicit definition need not. In general, statement of the minimality fo P requires infinitely many axioms, e.g. the circumscription schema. However, in the fortunate cases in which the explicit definition of P consists of a list l of possible values (as in the conjunctive block example above), the circumscription schema may be completely replaced by a finite axiomatization of P's minimality. Since I is finite, each of its proper subsets may be explicitly listed, and an axiom set $\mu(A,P,I)$ constructed which declares that each proper subset fails to satisfy $A(P)$. In this case, $\mathrm{Th}(K(A,P)) = \mathrm{Th}(A \cup \forall \vec{x}(P(\vec{x}) \equiv \phi(\vec{x})) \cup \mu(A,P,I))$.

Actually, while Beth's theorem is famous, it is never used in logic except as an excuse for ignoring implicit definitions, since it means that implicit definitions offer no greater expressiveness in defining functions than do explicit definitions. But Beth's theorem is an easy consequence of *Craig's Interpolation Lemma*, which is also famous, and which appears in many logical studies with many applications. This lemma states that if $\alpha \supset \beta$, then there is a formula γ involving only symbols common to both α and β such that $\alpha \supset \gamma$ and $\gamma \supset \beta$. That is, γ "interpolates" between α and β. For Beth's theorem, the explicit definition of P turns out to be a formula contained in the interpolant for $\hat{K}(A(P),P) \wedge \hat{K}(A(Q),Q) \supset \forall \vec{x}(P(\vec{x}) \equiv Q(\vec{x}))$: a formula $\phi(\vec{x})$ not involving either P or Q such that $\hat{K}(A(P),P) \wedge \hat{K}(A(Q),Q) \supset \forall \vec{x}(P(\vec{x}) \equiv \phi(\vec{x}) \equiv Q(\vec{x}))$. Craig's lemma has appeared in the artificial intelligence literature at least once. Gumb [1978] employs it explicitly as a means for pinpointing (perhaps too sharp a connotation) the source of inconsistencies between databases. That is, if some new piece of information S is inconsistent with a previous theory T, then $T \supset \neg S$, and there is an interpolant R with $T \supset R$ and $S \supset \neg R$, such that R contains only symbols common to both the old and new information. How specifically this identifies the "source" of the inconsistency is unclear: for in general, one might expect that all symbols in S appear in T, that the main interpolant is $\neg S$. A somewhat related idea is the "intelligent backtracking" devised by Pereira and Porto [1979] for PROLOG, in which the backtracker looks for shared and independent variables.

Even if it is little used in logic, Beth's theorem is of practical interest for artificial intelligence, since it yields a procedure for finding explicit definitions if they exist. The conceptually simplest procedure in our case is to enumerate the consequences of $K(A,P)$ and watch for one of the desired form. But there is a better way. Instead of this pure enumeration, we can use the first formulation of implicit definability and try to prove $\forall \vec{x}(P(\vec{x}) \equiv Q(\vec{x}))$ from $K(A(P),P)$ and $K(A(Q),Q)$ for some Q not in $\mathcal{L}$. A proof will only exist if $K(A,P)$ implicitly defines P, and an explicit definition can be extracted from the proof. Some proof systems (see [Smullyan 1968]) even carry the explicit definition along as an interpolant. Needless to say, this method is much more directed than the pure enumeration. Unfortunately, neither method can guarantee results. By the completeness of the predicate calculus, if an explicit definition exists, these methods will find one. But if an explicit definition does not exist, then we see that the methods will run on forever, searching in vain for the desired definition or for a non-existent proof. Worse, there is no way to modify the procedures to check first that an explicit definition exists. Whether a theory explicitly (or implicitly) defines a predicate is undecidable, so the only way to be sure an explicit definition does not exist is to search all the consequences. (Proof: We can translate questions about satisfiability into questions about implicit definitions by considering a formula like $\psi : \phi \vee [\neg \phi \wedge \forall x.P(x) \equiv x = x]$. If ψ implicitly defines P, we know that ϕ is not satisfiable, for if it were, P could be anything. And if ϕ is not satisfiable, then ψ implictly (even explicitly) defines P. Hence ϕ is unsatisfiable iff ψ implicitly defines P. The undecidability of satisfiability means that implicit definition must also be undecidable.)

If implicit definition is undecidable, can we find decidable special cases? In particular, can we find conditions on $A(P)$ which guarantee that $K(A,P)$ implicitly defines P? One such result is that the circumscription of a theory $A(P)$ implicitly defines P if $A(P)$ is monotone in P. Monotone in P means that the theory is equivalent to one in which P occurs only positively. Theories monotone in P are important because they can be used to define monotone (non-decreasing) operators corresponding to P. Monotone operators in turn are the basis of inductive definitions, in which one identifies the least fixed point of the operator as the set inductively defined by the operator (see [Aczel 1977]). If the monotone operator corresponding to P has a least fixed point, then that least fixed point is the predicate implicitly defined by $K(A,P)$. This allows us to define the circumscription of an infinite theory A as the theory corresponding to the least fixed point of the P operator derived from A. Unfortunately, whether a theory is monotone in P is also undecidable, so we must again look for tractable special cases. In fact, *Horn* *in* P is a special case of monotone in P, so circumscription implicitly defines P in each consistent $A(P)$ that is Horn in P. This means that the Reiter-Clark result on predicate completion in Prolog can be strengthened to yield an explicit definition $\forall \vec{x}(P(\vec{x}) \equiv \phi(\vec{x}))$ such that ϕ does not mention P.

Disjunctive Definability

If circumscription fails to implicitly define its intended predicate, then no concise characterization of the predicate exists as an explicit definition. But the disjunctive block example above suggests that we might be willing to settle for something weaker as a concise characterization: namely, a *disjunctive definition* of the form

$$\forall \vec{x}(P(\vec{x}) \equiv \phi_1(\vec{x})) \vee \ldots \vee \forall \vec{x}(P(\vec{x}) \equiv \phi_k(\vec{x}))$$

which we might abbreviate as

$$\bigvee \forall \vec{x}(P(\vec{x}) \equiv \vec{\phi}(\vec{x})).$$

Clearly, explicit definability is a special case of disjunctive definability, the trivial case of exactly one disjunct ($k = 1$).

One can view a disjunctive definition as a classification of all models with respect to P—just as in the block example, models were classified into two categories: those in which $b1$ is the only block, and those in which $b2$ is the only block. In fact, if P is explicitly definable in each model of $K(A, P)$, then a disjunctive definition must exist. That is, even though $K(A, P)$ has infinitely many models, they fall into a finite set of equivalence classes with respect to P. To see this, consider adding to $K(A, P)$ sentences of the form $\neg \forall \vec{x}(P(\vec{x}) \equiv \phi(\vec{x}))$ for each expression ϕ not involving P. If $K(A, P)$ is consistent, then this extended theory must be inconsistent since each model of $K(A, P)$ will falsify one of the added sentences, so by compactness there is a finite inconsistent subset. Then $K(A, P)$ entails the disjunction of the negations of the added statements in this finite set.

If a disjunctive definition of P in $K(A, P)$ exists, by the completeness of the predicate calculus it will be provable from $K(A, P)$, so we can find it by enumerating the consequences of $K(A, P)$ and looking for formulas of the form $\bigvee \forall \vec{x}(P(\vec{x}) \equiv \vec{\phi}(\vec{x}))$. But not only is the existence of a disjunctive definition undecidable, whether a disjunctive definition of a given size k exists is also undecidable, and consequently there is no algorithm for computing the smallest disjunctive definition. In particular, even if we believe that a disjunctive definition exists, there is no way to tell from $K(A, P)$ how large any disjunctive definition must be except by finding some disjunctive definition and taking its size as an upper bound. Thus short of finding a small expression, we cannot be sure that the desired disjunctive definition does not have a million disjuncts. Moreover, for each k there are theories whose smallest disjunctive definition of P is of size k. Consider, for example, the axiom

$$\forall x(P(x) \equiv [x = 1]) \vee \ldots \vee \forall x(P(x) \equiv [x = k]).$$

(I suspect there are smaller theories with this property.) However, there are direct procedures for finding disjunctive definitions of specific sizes, procedures extending the interpolation-based procedure for finding explicit definitions. If we seek a definition of P in $A(P)$ with k disjuncts, we assume

$$\begin{aligned}
&A(P) \wedge A(P_1) \wedge \ldots \wedge A(P_k) \\
&\quad \wedge \neg \forall \vec{x}(P(\vec{x}) \equiv P_1(\vec{x})) \wedge \ldots \wedge \neg \forall \vec{x}(P(\vec{x}) \equiv P_{k-1}(\vec{x})) \\
&\quad \wedge \neg \forall \vec{x}(P_1(\vec{x}) \equiv P_2(\vec{x})) \wedge \ldots \wedge \neg \forall \vec{x}(P_1(\vec{x}) \equiv P_k(\vec{x})) \\
&\quad \ldots \\
&\quad \wedge \neg \forall \vec{x}(P_{k-1}(\vec{x}) \equiv P_k(\vec{x}))
\end{aligned}$$

and try to prove $\forall \vec{x}(P(\vec{x}) \equiv P_k(\vec{x}))$. If we succeed, we extract an explicit formula ϕ_k for P_k and iterate, seeking this time a $k-1$ fold disjunctive definition for P in $A(P) \wedge \neg \forall \vec{x}(P(\vec{x}) \equiv \phi_k(\vec{x}))$. When we finish, if ever, we know that

$$A(P) \vdash \forall \vec{x}(P(\vec{x}) \equiv \phi_1(\vec{x})) \vee \ldots \vee \forall \vec{x}(P(\vec{x}) \equiv \phi_k(\vec{x})).$$

Definability with parameters

Logicians have also studied the notions of explicit definability up to parameters and disjunctive definability up to parameters. These mean, respectively, the existence of formulas ϕ, ϕ_1, $\ldots$, ϕ_k of arities $m = n + l$ such that for $\vec{z} = z_1, \ldots, z_l$

$$A \vdash \exists \vec{z} \forall \vec{x}(P(\vec{x}) \equiv \phi(\vec{x}, \vec{z}))$$

and

$$A \vdash \exists \vec{z} \forall \vec{x}(P(\vec{x}) \equiv \phi_1(\vec{x}, \vec{z})) \vee \ldots \vee \exists \vec{z} \forall \vec{x}(P(\vec{x}) \equiv \phi_k(\vec{x}, \vec{z})).$$

These definitions express P as projections of explicitly defined formulas. I am not sure if these expressions can be of as much use in artificial intelligence as the unparameterized forms.

Completions

Another way of looking at circumscription is as an attempt to find a minimal completion of a theory with respect to a predicate, rather than as an implicit definition. In general, the axioms $A(P)$ will entail only some ground cases of P: sentences of the form $P(\vec{c})$ and $\neg P(\vec{c})$ for some constants $\vec{c}$. A will leave P undecided for other constant tuples, but may restrict their possibilities by entailing disjunctions of the form $\pm P(\vec{c}_1) \vee \ldots \vee \pm P(\vec{c}_k)$. To interpret these axioms as complete information, we must extend A so that the extension decides all ground instances of P. By focussing on "positively minimal" completions which make P hold for as few tuples as possible, we see that one of two cases concerning completions must be true of A. Let $C(A, P)$ be the set of negative ground instances of P for all the instances undecided by A, that is,

$$C(A, P) = \{\neg P(\vec{c}) \mid A \not\vdash P(\vec{c}) \wedge A \not\vdash \neg P(\vec{c})\}.$$

Let $D(A, P) = A \cup C(A, P)$. Then either $C(A, P)$ is consistent with A, in which case $D(A, P)$ is the desired extension of A, or it is inconsistent with A. If $D(A, P)$ is inconsistent, it must be that A entails some disjunctions of the form $P(\vec{c}_1) \vee \ldots \vee P(\vec{c}_k)$. In this case, every maximal subset of $C(A, P)$ consistent with A is a completion, and is as good a completion as any other, as far as logic is concerned.

Extending a theory A to $D(A, P)$ is usually expressed as making the ***closed world assumption*** about P in A. While the motivations behind circumscription and the closed world assumption are the same, formally they are almost completely unrelated. $K(A, P)$ says only that P is minimal among those predicates satisfying $A(P)$, but may leave ground cases undecided. That is, in general there will be many ground instances $P(\vec{c})$ such that $K(A, P) \not\vdash P(\vec{c})$ and $K(A, P) \not\vdash \neg P(\vec{c})$. Conversely, $D(A, P)$ determines P only for those ground instances expressible in the language L, and says nothing about P's behavior on unnamed elements of the domain. In particular, $D(A, P)$ does not require P to be minimal in any way. It would be interesting to know if $D(K(A, P), P)$ has any useful properties. But in this connection, it is worth noting an important theoretical difference between $K(A, P)$ and $D(A, P)$, namely that $K(A, P)$ is recursively enumerable (assuming A is), while $D(A, P)$ is in general not recursively enumerable. In practice, however, the closed world assumption is useful for axiom sets A of certain forms, for example when A is Horn in P. In such cases, the closed world assumption always preserves the consistency of A, and fits well with some procedures for automated deduction. (See [Reiter 1978, 1980, 1982].)

For the case in which the closed world assumption produces an inconsistent extended theory, Minker [1982] has defined the *generalized closed world assumption* to extend A with all the ground cases (positive and negative) common to all the consistent completions of A with respect to P.

Lipski [1977, 1983] has studied the notions of external and internal interpretations of databases. The internal interpretation appears related to the closed world assumption, and admits characterizations in terms of topological boolean algebras and modal logics. It would be nice to know more about the relation between this internal interpretation and circumscription.

Conservation questions

If circumscription is to be routinely useful in artificial intelligence practice, then it is worth knowing how new axioms affect circumscriptively obtained results. Let $A' \supseteq A$ be an extended set of axioms. It is easy to see that A' may implicitly define P even if A does not. For instance, let A be the axioms of the disjunctive block example, and A' these axioms plus the conjunctive example axioms. Since the conjunctive axioms entail the disjunctive ones, the resulting theory has the same conclusions as the conjunctive example axioms alone, and these implicitly define the predicate *Block*. Conversely, A may implicitly define P but A' may not. For instance, let A be the conjunctive block axioms, and A' be these plus $\mathit{Block}(b3) \vee \mathit{Block}(b4)$. And of course, circumscription is non-monotonic in that $K(A, P)$ need not entail $K(A', P)$, and $K(A', P)$ need not entail $K(A, P)$. Indeed, in many cases of interest, these sets will be inconsistent. For instance, augmenting the conjunctive block axioms with a third block $b3$ yields a theory inconsistent with the two-block circumscription. Unfortunately, conservation questions are largely unstudied. We take the present opportunity to ask several obvious ones.

First, if A entails A', then clearly $K(A, P)$ implicitly defines P iff $K(A', P)$ does too, and moreover $\mathrm{Th}(K(A, P)) = \mathrm{Th}(K(A', P))$. In some cases, the same is true even though A does not entail A', for example, if A' is A plus the explicit definition of P in $K(A, P)$. But when, precisely, does $K(A, P)$ entail $K(A', P)$? If and only if $A' \subseteq \mathrm{Th}(K(A, P))$?

When are new axioms irrelevant to prior circumscriptions? That is, when is $\mathrm{Th}(K(A', P)) = \mathrm{Th}(A' \cup K(A, P))$?

Finally, if $K(A, P)$ fails to implicitly define P, can one characterize extensions A' which do implicitly define P? What is possible if we require $A' - A$ to be finite?

Language extensions

Even if implicit definability fails, there is sometimes still an explicit definition—but in an extended language, in which the extended explicit definition can rule out "non-standard" interpretations. For example, Gödel showed how truth in arithmetic may be "implicitly defined" in arithmetic, yet lack an explicit definition within arithmetic, while Tarski showed how a simple extension to the language of arithmetic permits an explicit definition of truth in arithmetic (but not truth in the extended language). There are many questions that may be interesting here. For instance, if $K(A, P)$ fails to implicitly define P, when can some simple extension to $\mathcal{L}$ permit an explicit definition? And how can it be found? Is it practical to use this idea in the meta-theoretical systems popular in artificial intelligence? Is it possible to adapt the ideas of Kripke's [1975] theory of truth to implicit definitions of other predicates?

Conclusion

We have surveyed the motivations for circumscription and its connections with logical notions like implicit definability. I hope that some of the questions raised above will prove as interesting to the logician as some of the logical techniques may prove to the artificial intelligence researcher, for much work remains to be done. We recall the main sorts of open problems:

1. What are important cases in which $K(A, P)$ produces explicit definitions or small disjunctive definitions? How large are these definitions relative to the size of A? Can these special cases be recognized, and are they of common importance (e.g. Horn databases)? What is the cost of finding these definitions given their existence?

2. What does circumscription do when it fails to implicitly define P? Can its consequences be characterized in some interesting way?

3. How should revision of circumscriptive conclusions be mechanized?

One important topic slighted above is the analog of circumscription for systems other than logical languages. For example, in the abstract, circumscription can be applied in all sorts of representational systems by formulating data-structures via inductive definitions and looking for least fixed points or least solutions. See [Aczel 1977] for a treatment of inductive definitions in general, and [Scott 1982] for a "propositional" treatment of data-structures.

Many have wondered about the relation between circumscription and non-monotonic logic. It seems fairly clear now that there is little relation. The preceding shows that circumscription is fundamentally a *logical* topic, the study of minimal solutions to axiomatized predicates. [Doyle 1982] shows that the reasoned assumptions appearing in reason maintenance systems and in non-monotonic logic are preferences of the agent concerning its own "state of mind," and that they comprise a fundamentally *psychological* or *decision theoretic* topic, the study of value and choice in the mental operations of the agent.

We close by noting that little is known about when and how to use circumscription in reasoning and decision-making. But that is not a problem about the logical nature or mechanization of circumscription, and we leave it to future artificial intelligence research.

References

For accounts of Beth's theorem, see Chang and Kiesler pp. 87-88; Mostowski pp. 127-128; Shoenfield p. 81; and Smullyan pp. 131-133. For disjunctive definability and parameterized versions, see Chang and Kiesler pp. 250-259. For interpolation algorithms, see Smullyan.

Aczel, P., 1977. An introduction to inductive definitions, in *Handbook of Mathematical Logic* (J. Barwise, ed.), Amsterdam: North-Holland, 739-782.

Chang, C. C., and Kiesler, H. J., 1973. *Model Theory*, Amsterdam: North-Holland, 87-88, 250-259.

Clark, K. L., 1978. Negation as failure, *Logic and Data Bases* (H. Gallaire and J. Minker, eds.), New York: Plenum Press, 293-322.

Davis, M., 1980. The mathematics of non-monotonic reasoning, *Artificial Intelligence* **13**, 73-80.

Doyle, J., 1982. Some theories of reasoned assumptions: an essay in rational psychology, Pittsburgh: Department of Computer Science, Carnegie-Mellon University.

Gumb, R. D., 1978. Computational aspects of evolving theories, *SIGART Newsletter* No. 67, 13.

Kripke, S. A., 1975. Outline of a theory of truth, *Journal of Philosophy* **72**, 690-716.

Lipski, W., 1977. On the logic of incomplete information, *Proc. 6th International Symposium on the Mathematical Foundations of Computer Science*, Lecture Notes in Computer Science No. 53, Berlin: Springer-Verlag, 374-381.

Lipski, W. 1983. Logical problems related to incomplete information in databases, Laboratoire de Recherche en Informatique, Universite de Paris-sud, Report 138.

McCarthy, J., 1977. Epistemological problems of artificial intelligence, *Proc. Fifth International Joint Conference on Artificial Intelligence*, 1038-1044.

McCarthy, J., 1980. Circumscription—a form of non-monotonic reasoning, *Artificial Intelligence* **13**, 27-39.

Minker, J., 1982. On indefinite databases and the closed world assumption, *Proc. 6th Conference on Automated Deduction* (D. Loveland, ed.), Berlin: Springer-Verlag, 292-308.

Mostowski, A., 1966. *Thirty Years of Foundational Studies*, New York: Barnes and Noble, 127-128.

Pereira, L. M., and Porto, A., 1979. Intelligent backtracking and sidetracking in Horn clause programs—the theory, Departamento de Informatica, Universidade Nova de Lisboa, Report 2/79.

Reiter, R., 1978. On closed world data bases, in *Logic and Data Bases* (H. Gallaire and J. Minker, eds.), New York: Plenum Press, 55-76.

Reiter, R., 1980. A logic for default reasoning, *Artifical Intelligence* **13**, 81-132.

Reiter, R., 1982. Circumscription implies predicate completion (sometimes), *AAAI-82*, 418-420.

Scott, D. S., 1982. Domains for denotational semantics, *Proc. ICALP-82*.

Shoenfield, J. R., 1967. *Mathematical Logic*, Reading: Addison-Wesley, 81.

Smullyan, R. M., 1968. *First-Order Logic*, Berlin: Springer-Verlag, Ch. XV, §2, 131-133.