On Reducing Parallel Circumscription

Li Yan Yuan and Cheng Hui Wang
The Center for Advanced Computer Studies
University of Southwestern Louisiana
Lafayette, LA 70504

Abstract

Three levels of circumscription have been proposed by McCathy to formalize common sense knowledge and non-monotonic reasoning in general-purpose database and knowledge base systems. That is, basic circumscription, parallel circumscription, and priority circumscription. Basic circumscription is a special case of parallel circumscription while parallel circumscription is a special case of priority circumscription. Lifschitz has reduced priority circumscription into parallel circumscription, i.e., represented priority circumscription as a conjunction of some parallel circumscription formulas. In this paper, we have reduced parallel circumscription into basic circumscription under some restriction, i.e., parallel circumscription of a Z-conflict free first order logic formula can be represented as a conjunction of some basic circumscription formulea.

1. Introduction

McCarthy has proposed circumscription to formalize common sense knowledge and non-monotonic reasoning designated to handle incomplete and negative information in database and knowledge base systems [McCarthy, 1980, McCathy, 1986]. Different levels of circumscription have been proposed for different kinds of application [McCathy, 1986, Lifschitz, 1985]. Assume A(P, Z) is a first order theory, where P and Z are disjoint sets of predicates in A. *Parallel circumscription*, denoted as CIR(A; P; Z), asserts that the extension of P should be minimized under the condition of A(P; Z), while Z is allowed to vary. When $Z = \emptyset$, parallel circumscription reduces to CIR(A; P), which we call *basic circumscription*[1]. Minimizing a set P of predicates may conflict with each other. Thus, *priority circumscription* has been proposed. Priority circumscription, CIR(A; $P^1 > P^2 > \ldots > P^n$; Z), where $P^1, \ldots, P^n$, Z are pairwise disjoint sets of predicates in A, expresses the idea that predicates in P^1 should be minimized at higher priority than those in P^2, P^2 at higher priority than those in P^3, etc, while Z is allowed to vary.

Obviously, basic circumscription is a special case of parallel circumscription, and parallel circumscription is a special case of priority circumscription. Lifschitz has reduced priority circumscription into parallel circumscription, i.e., a priority circumscription can be represented by a conjunction of some parallel circumscription formulae [Lifschitz, 1985]. He has also tried to reduce parallel circumscription into basic circumscription. However, as he indicated, the result is not satisfactory, since a second-order quantifier is introduced within circumscription.

Przymusinski has proposed an algorithm to compute parallel circumscription, under certain assumptions [Przymusinski]. Because of the difficulties brought in by Z, the algorithm has to treat parallel circumscription and basic circumscription separately, and the complexity for parallel circumscription is much higher than for basic circumscription. If we could reduce parallel circumscription into basic circumscription, his algorithm can be simplified dramatically and be much more efficient.

Therefore, from both the theoretical and practical points of view, we would like to reduce parallel circumscription into basic circumscription, if possible.

In this paper, we first define the Z-resolution process, which is used to transfer all negative literals of Z into positive ones, without lossing of logical connection between other predicates. If the Z-resolution successes, then we are able to eliminate all rules which contain predicate Z without affecting computing parallel circumscription. Then a class of first order theory, called Z-conflict free, is defined. When the given theory is Z-conflict free, an algorithm is presented to eliminate all Z predicates from A. Finally, we show that when the given theory is Z-conflict free, parallel circumscription can be reduced

1 In the literature, parallel circumscription with empty Z is usually used. However, for the sake of clarity, the term basic circumscription is used here instead.

into basic circumscription.

The rest of this paper is organized as follows. In Section 2, we recall the definition of circumscription and some preliminary results. In Section 3, some properties about logical systems are discussed. Section 4 shows how Z-resolution can be used to reduce parallel circumscription into basic circumscription. In Section 5, we show that the restriction can be removed in many cases.

2. Preliminary Results

In this section, we briefly discuss some fundamental concepts and preliminary results which are useful for the following discussion.

There are three kinds of circumscription as formalized in [Lifschitz, 1985].

Basic Circumscription Let A be a first order logic formula, $P = \{ P_1, \ldots, P_n\}$ be a set of predicates in A. The basic circumscription of P in A, denoted as CIR(A; P), is a second-order formula

$$A(P) \wedge \neg \exists P'(A(P') \wedge P' < P),$$

where P' is a tuple of predicate variables similar to P, and $P' < P$ means

$$\bigwedge_{i=1}^{n} \forall \mathbf{x}\, (P_i'(\mathbf{x}) \supset P_i(\mathbf{x})) \wedge \bigvee_{i=1}^{n} \exists \mathbf{x}\, (P_i(\mathbf{x}) \wedge \neg P_i'(\mathbf{x})),$$

where **x** is a tuple of variables.

Parallel Circumscription Let A(P, Z) be a first order logic formula, where $P = \{P_1, \ldots, P_n\}$ and $Z = \{Z_1, \ldots, Z_m\}$ are two disjoint sets of predicates in A. The circumscription of P in A(P, Z) with variable Z, denoted as CIR(A; P; Z), is a second order formula

$$A(P, Z) \wedge \neg \exists P', Z'(A(P', Z') \wedge P' < P),$$

where P', Z' are tuples of predicate variables similar to P and Z, and $P' < P$ has the same meaning as above.

Priority Circumscription Let $A(P^1, P^2, \ldots, P^n, Z)$ be a first order formula, where $P^i = \{ P_{i_1}, \ldots, P_{i_{kj}} \}$, $i = 1, \ldots, n$, and $Z = \{ Z_1, \ldots, Z_m\}$ are pairwise disjoint sets of predicates in A. The priority circumscription of A, denoted as $CIR(A; P^1 > P^2 > \ldots > P^n; Z)$, is defined as a second order formula

$$A(P, Z) \wedge \neg \exists P', Z' (A(P', Z') \wedge P' \approx P)$$

where, $P = \{P^1, \ldots, P^n\}$, and P' and Z' stands for P and Z, and $P' \approx P$ means $\bigwedge_{i=1}^{n} (\bigwedge_{j=1}^{i-1} P^{j\prime} = P^j \supset P^{i\prime} \leq P^i) \wedge P' \neq P$, and $P^{i\prime} \leq P^i$ means $P_i' < P_i$ or $P_i' = P_i$.

Lifschitz has tried to reduce parallel circumscription into basic circumscription, as stated below.

Theorem 2.1 [Lifschitz, 1985] $CIR(A(P, Z); P; Z) \equiv A(P, Z) \wedge CIR(\exists Z'\, A(P, Z'); P)$. □

As noticed by Lifschitz, theorem 2.1 does not strip off Z in circumscription, since the formula contains a second-order quantifier. However, Lifschitz has successfully reduced priority circumscription to parallel circumscription, as shown in the following theorem .

Theorem 2.2 [Lifschitz, 1985] $CIR(A; P^1 > P^2 > \ldots > P^k; Z) \equiv \bigwedge_{i=1}^{k} CIR(A; P^i ; P^{i+1}, \ldots, P^k, Z)$. □

3. Z-Recursion and One-Side Predicates

Let A be a first order formula. Without loss of generality, we assume A is in clausal form, i.e., a set of clauses. Each clause r in A has the form

$$\neg Q_1 \vee \neg Q_2 \vee \ldots \neg Q_m \vee P_1 \vee P_2 \vee \ldots \vee P_n,$$

where Q_i, P_j are predicates and may contain variables. A clause r may be rewritten in the form of

$$Q_1 \wedge \ldots \wedge Q_m \supset P_1 \vee \ldots \vee P_n,$$

and is called a *rule* in A.

Given a rule r, LHS(r) is used to denote the set of all predicates occurring negatively in r, and RHS(r) the set of all predicates occurring positively in r.

Recursion plays an important role in logical system implementation. Since we are interested in computing parallel circumscription CIR(A; P; Z), we discuss only the recursion associated with the set Z of predicates in A.

Let A be a first order formula, Z be the set of predicates in A. A binary relation is defined on Z. Assume Z_i, Z_j are two predicates in Z (Z_i and Z_j are not necessarily distinct), then we say Z_i *derives* Z_j, denoted as $Z_i \rightarrow Z_j$, if there exists a rule r in A such that $Z_i \in LHS(r)$ and $Z_j \in RHS(r)$. We define $\rightarrow^*$ to be the transitive closure (not the reflexive transitive closure) of $\rightarrow$.

Z_i and Z_j are *mutually Z-recursive* if $Z_i \rightarrow^* Z_j$ and $Z_j \rightarrow^* Z_i$. Z_i is *Z-recursive* if $Z_i \rightarrow^* Z_i$. Otherwise Z_i is *Z-recursion free*. It can be easily shown that mutual Z-recursion is an equivalence relation on the set of Z-recursive predicates [Bancilhon *et al.*, 1986].

A rule r in A is said to be *Z-recursive* if there exist two predicates Z_i and Z_j in Z, (Z_i and Z_j are not necessarily distinct) such that $Z_i \in LHS(r)$, $Z_j \in RHS(r)$, and Z_i and Z_j are mutually recursive. Otherwise, r is Z-recursion free.

A predicate Z_i is said to be Z-recursion free in a rule r if $Z_i \in RHS(r)$ and for each $Z_j \in LHS(r)$, Z_i and Z_j are not mutually recursive. The fact that Z_i is Z-recursion free in r does not necessarily imply that Z_i is Z-recursion free in A.

Example 3.1 Assume A is given by the following rules:

r_1: $Q_1 \supset Z_1 \vee P_1$
r_2: $Z_1 \supset Z_2 \vee Q_2$
r_3: $P_2 \wedge Z_2 \supset Z_1 \vee Q_3 \vee Z_3$
r_4: $Q_2 \wedge Z_3 \supset P_2$
r_5: $Q_3 \supset Z_3$.

Then, $Z_1 \rightarrow Z_2$, $Z_2 \rightarrow Z_1$ and $Z_2 \rightarrow Z_3$. Let $Z = \{Z_1, Z_2, Z_3\}$. Thus Z_1 and Z_2 are mutually Z-recursive. Z_3 is Z-recursion free, Z_1 is Z-recursion free in r_1, and Z_3 is Z-recursion free in r_3 and r_5. r_2 and r_3 are Z-recursive, while r_1, r_4, and r_5 are Z-recursion free. □

The Z-recursion is defined regardless of the terms occurring in predicates. Thus, $Z_1(x) \supset Z_1(a)$ is Z-recursive. The reason is that such a definition has no impact on our implementation method, but simplifies our discussion.

Now, we discuss a technique used to simplify computing circumscription.

Consider Example 3.1. If we assume both Z_1 and Z_2 are true, then r_1, r_2 and r_3 are always satisfied. Because Z is allowed to vary, such an assumption is valid. Therefore, in the processing of minimizing P when we compute CIR(A; P; Z), r_1, r_2 and r_3 make no contribution, so they can be deleted. Let A′ contain only r_4 and r_5, then it is easy to show that

$CIR(A; P; Z) \equiv CIR(A'; P; Z_3) \wedge A(P, Z) \equiv (P_1 \equiv \text{false}) \wedge (P_2 \supset Q_2 \wedge Q_3) \wedge A(P, Z)$.

Motivated by the above example, we propose the one-side predicate as defined below.

Definition 3.1 Let A be a first order formula, Z be a set of predicates. Let $z \subseteq Z$ be a set of predicates. z is said to be ***left-side*** if for each r in A, either $RHS(r) \cap z = \emptyset$ or $LHS(r) \cap z \neq \emptyset$. z is ***right-side*** if for each r in A, either $LHS(r) \cap z = \emptyset$ or $RHS(r) \cap z \neq \emptyset$. z is ***one-side*** if z is either left-side or right-side. □

In Example 3.1, $\{Z_1, Z_2\}$ is right-side. The significance of defining the one-side predicate is demonstrated by the following theorem.

Theorem 3.1 Let A(Q, P, Z) be a first order formula, z be a one-side set of predicates in Z. A′(Q, P, Z) be a formula obtained from A by deleting all rules containing some predicates in z. Then $CIR(A; P; Z) \equiv CIR(A'; P; Z) \wedge A(P, Z)$. □

Theorem 3.1 can be used to simplify computing CIR(A; P; Z). However, unless Z is entirely one-side, we cannot avoid computing parallel circumscription.

4. Z - Resolution

In this section, we first present an algorithm, called *Z-resolution*, to simplify the given theory, and then show that under certain condition, the Z-resolution can be used to reduce parallel circumscription into basic one. Like the Robinson resolution, the idea of Z-resolution is very simple as demonstrated below.

Example 4.1 Assume A is defined by the following two rules:

$Q_1(x) \supset Z(x) \vee P(x)$ (1)
$Z(x) \supset Q_2(x).$ (2)

Then, if we replace Z(x) in the first clause by $Q_2(x)$, we have:

$Q_1(x) \supset Q_2(x) \vee P(x).$ (3)

Let A′ contain (3), then it is easy to show that:

$CIR(A(Q, P, Z); P; Z) \equiv CIR(A'(Q, P); P) \wedge A(Q, P, Z)$. □

This example motivates us trying to eliminate all Z predicates from A, while still remain logical connection between those predicates in Q and P.

Let us briefly discuss some notations. A set of expressions $\{ \Phi_1, ..., \Phi_n \}$ is unifiable if and only if there is a ***substitution*** σ that makes the expressions identical. In such a case, σ is said to be a ***unifier*** for that set. A ***most general unifier*** γ of Φ and Ψ has the property that, if σ is any unifier of the two expressions, then, there exists a substitution δ with the following property:

$\Phi\gamma\delta = \Phi\sigma = \Psi\sigma.$

If a subset of the literals in a clause Φ has a most general unifier γ, then, the clause Φ' is called a ***factor*** of Φ if it is obtained by applying γ to Φ. Let Φ and Ψ are two clauses, if there is a literal $\neg\phi$ in some factor Φ' of Φ and a literal ψ in some factor Ψ' of Ψ such that Φ and Ψ have a most general unifier γ, then the clause $(\Phi' - \{\Phi\}) \cup (\Psi' - \{\neg\Psi\})\gamma$ is called a resolvent of the two clauses using Φ[Genesereth *et al.*, 1987]. In Example 4.1, (3) is a resolvent of (1) and

(2) using Z(x).

Let r_1 and r_2 be two clauses, $\neg\phi$ be a literal in r_1, $\psi_1, \psi_2, \ldots, \psi_n$ be all literals in r_2 that have most general unifiers with ϕ. $S_1, S_2, \ldots, S_n$ is a sequence resolvents of r_1 and r_2 using ϕ. That is S_1 is the resolvent of r_1 and r_2 using ϕ, S_2 the is the resolvent of r_2 and S_1, ..., etc. Then the ϕ-resolvent of r_1 and r_2 using ϕ is defined as S_n.

Example 4.2 Let

r_1: $Q_1(x, y) \supset Z(x, y) \vee Z(y, x)$

r_2: $Z(x, y) \wedge Q_2(x, y) \supset P(x, y)$.

Then, the Z-resolvent of r_2 and r_1 using Z(x, y) is the clause

$$Q_1(x, y) \wedge Q_2(x, y) \wedge Q_2(y, x) \supset P(x, y) \vee P(y, x).$$

□

Let A be a set of clauses, Φ be a clause in A, $\neg Z$ be a negative literal in Φ, $A'(\Phi, Z)$ be the set of all Z-resolvents of Φ with each clause in A which contains positive occurrence from Z. Then the Z-resolution set $R(A, \Phi, Z)$ is defined as $A'(\Phi, Z) \cup (A - \Phi)$.

Lemma 4.1 $R(A, Z) \equiv A$. □

Example 4.3 Assume A contain the following clauses:

r_1: $Q_1(x, y) \supset Z_1(x, y) \vee Z_2(x, y)$

r_2: $Z_2(x, y) \supset Q_2(x, y) \vee P_1(x, y)$

r_3: $Z_1(x, y) \wedge Z_1(y, z) \supset P_2(x, z)$

Then, $A_1 = R(A, r_3, z_1(x, y))$ contains:

r_1: $Q_1(x, y) \supset Z_1(x, y) \vee Z_2(x, y)$

r_2: $Z_2(x, y) \supset Q_2(x, y) \vee P_1(x, y)$

r_4: $Q_1(x, y) \wedge Z_1(y, x) \supset P_2(x, z) \vee Z_2(x, y)$.

$A_2 = R(A_1, r_4, Z_1(y, x))$ contains:

r_1: $Q_1(x, y) \supset Z_1(x, y) \vee Z_2(x, y)$

r_2: $Z_2(x, y) \supset Q_2(x, y) \vee P_1(x, y)$

r_5: $Q_1(x, y) \wedge Q_1(z, x) \supset P_2(z, y) \vee Z_2(z, x) \vee Z_2(x, y)$. □

By examing A_2, we find that Z_1 becomes one side predicate in A_2. As far as computing parallel circumscription is concerned, we may obtain an A_3 from A_2 by deleting r_1. That is A_3 contains only r_2 and r_5.

Let $A_4 = R(A_3, r_2, Z(x, y))$. Then A_4 contains:

r_5: $Q_1(x, y) \wedge Q_1(z, x) \supset P_2(z, y) \vee Z_2(z, x) \vee Z_2(x, y)$

r_6: $Q_1(x, y) \wedge Q_1(z, x) \supset P_2(z, y) \vee Q_2(z, x)\ P_1(z, x) \vee Q_2(x, y) \vee P_1(x, y)$.

Since r_5 is one side in A_4, $A_5 = \{r_6\}$. Then, by Theorem 3.1,

$$CIR(A; P; Z) \equiv CIR(A_6; P) \wedge A(Q, P, Z).$$

Given a theory A, the Z-resolution tries to transfer all negative occurrences of Z into positive ones, i.e., one side. If the process successes, by Theorem 3.1, the parallel circumscription can be reduced into basic circumscription. Unfortunately, the process may not always success.

Example 4.4 Let A contain only two rules as follows:

r_1: $Q(x) \supset Z(x, y) \vee Z(y, x)$

r_2: $Z(x, y) \wedge Z(y, x) \supset P(x, y)$.

Then we simply can not transfer Z into one side by Z-resolution.

Now, we specify a class of theories for which the Z-resolution guarantees the reducing of parallel circumscription into basic one.

Let A be a set of clauses, and Z be a set of predicate symbols in A. A binary relation is defined on Z as follows. Assume Z_i, Z_j are two predicates in Z, then $Z_i \Rightarrow Z_j$ if either $Z_i \rightarrow Z_j$, or there exists an predicate Z_k from Z and two clauses r_1 and r_2 in A such that $\{Z_i, Z_k\} \in LHS(r_1)$ and $\{Z_j, Z_k\} \in RHS(r_2)$. We define $\Rightarrow^*$ to be the transitive closure (not the reflexive closure) of $\Rightarrow$. Z_i and Z_j are extended Z-recursive if $Z_i \Rightarrow^* Z_j$ and Z_j =-0.1'>*. Z_i is extended Z-recursive if $Z_i \Rightarrow^* Z_i$. Extended Z-recursion is an equivalence relation of the set of extended Z-recursive predicates.

Let A be a set of clauses and Z be a set of predicates. A is said to be Z-conflict free if whenever there exist a clause r, and two predicates Z_i and Z_j such that $\{Z_i, Z_j\} \in LHS(r)$, then Z_i and Z_j are not extended Z-recursive. A in Example 4.3 is Z-conflict free, while A in Example 4.4 is not.

Let A be a set of clauses and Z be a set of predicates in A. An SP-ordering of Z is defined as an sequence $Z_1, Z_2, \ldots, Z_n$ such that $i < j$ implies that if $S_j \Rightarrow^* S_i$, then $S_i \Rightarrow^* S_j$.

An SP-ordering of Z always exists, though it may not be unique.

Now we present an algorithm to reduce parallel circumscription of A into basic circumscription when A is Z-conflict free.

Function REDUCE (A; Z);

Input: A Z-conflict free set A(Q, P, Z) of clauses.

Output: REDUCE(Q, P) such that CIR(A; P; Z) $\equiv$ CIR(REDUCE; P) $\wedge$ A(Q, P, Z).

Method:

```
begin
  Let Z1, Z2, ..., Zn be an SP-ordering of Z;
  for i = 1 step 1 to n do
    begin
      repeat
        select a clause r from A such that Zi ∈ LHS(r)
        and RHS(r) ∩ Zi = ∅;
        Let ¬Zi from Z be an negative literal in r;
        A := R(A; r, Zi);
      until Zi is one side in A;
      delete all clauses which contain Zi from A;
    end
    REDUCE := A
end.
```

Theorem 4.1 If A(Q, P, Z) is Z-conflict free, then CIR(A; P; Z) $\equiv$ CIR(REDUCE; P) $\wedge$ A(Q, P, Z). □

5. Further Discussion

Given a Z-conflict free theory A, the parallel circumscription of A can be reduced into basic one by Z-resolution. However, we are also able to transform many Z-conflict theories into Z-conflict free theories without affecting the result of circumscription. Let A be a set of clauses, Z be a predicate in A. Z is said to be negated if all positive literals from Z are changed into negative, and vice versa. Assume A′ is a first order formula obtained from A by negating some z from Z in A, the circumscription models for A and A′ differ only with the assignments of the z which have been negated. The following example shows how this method works.

Example 5.1 Let A contain two rules:

$Z_1(x, y) \wedge Z_2(y, x) \supset P_1(x, y)$
$Q_1(x, y) \supset Z_1(x, y)$
$Q_2(x, y) \supset Z_2(x, y)$
$Z_1(x, y) \supset Z_2(x, y) \vee Q_3(x, y)$
$Z_2(x, y) \supset Z_1(x, y) \vee Q_4(x, y)$

Obviously A is not Z-conflict free. By negating Z_2, we obtain a Z-conflict free theory A′ containing the following two rules:

$Z_1(x, y) \supset Z_2(y, x) \wedge P_1(x, y)$
$Q_1(x, y) \supset Z_1(x, y)$
$Q_2(x, y) \wedge Z_2(x, y) \supset F$
$Z_1(x, y) \wedge Z_2(x, y) \supset Q_3(x, y)$
$Z_2(x, y) \wedge Z_1(x, y) \supset Q_4(x, y)$

Following lemma demonstrates the significance of this transformation.

Lemma 5.1 Let A(Q, P, Z) be a set of clauses, A′ be a set of clauses obtained from A by negating a subset z from Z. Then

$$\mathrm{CIR}(A(Q, P, Z); P; Z) \equiv \mathrm{CIR}(A'; P; Z') \wedge \bigwedge_{z_i \in z} \forall x(Z_i(x) = \neg Z_i'(x)).$$ □

However, not all Z-conflict theories can be transformed to Z-conflict free theories by negating. A notable example is A in Example 4.4.

References

[Bancilhon *et al.*, 1986] Bancilhon, F. and Ramakrishnan, R., An Amateur's Introduction to Recursive Query Processing Strategies, Proc. of ACM SIGMOD, 1986, pp. 16-51.

[Clark, 1978] Clark, K., Negation as Failure, In: Logic and Database (Gallaire, H. and Minker, J., Eds.), Plenum Press, N. Y., 1978, pp. 293-322.

[Genesereth *et al.*, 1987] Genesereth, M. and Nilsson, N., Logical Foundation of Artificial Intelligence, Morgan Kaufmann Pub., Inc. Los Altos, CA, 1987.

[Lifschitz, 1985] Lifschitz, V., Computing Circumscription, Proc. of the 9th Int. Joint Conf. on AI, 1985, pp. 121-127.

[McCarthy, 1980] McCarthy, J., Circumscription - A Form of Non-Monotonic Reasoning, Artificial Intelligence 13, 1980, pp. 295-323.

[McCathy, 1986] McCarthy, J., Application of Circumscription to Formalizing Commonsense Knowledge, AAAI Workshop on Non-Monotonic Reasoning, 1984, pp. 295-323.

[Przymusinski] Przymusinski, T.G., An Algorithm to Compute Circumscription, to appear in J. Artificial Intelligence.