Computing Circumscriptive Databases by Integer Programming: Revisited
(Extended Abstract)

Ken Satoh, Hidenori Okamoto
Division of Electronics and Information, Hokkaido University
N13W8 Kita-ku, Sapporo, 060-8628, Japan
Email:ksatoh@db-ei.eng.hokudai.ac.jp

Abstract
In this paper, we discuss a method of computing minimal models in circumscription using integer programming in propositional logic and first-order logic with domain closure axioms and unique name axioms. This kind of treatment is very important since this enables us to apply various techniques developed in operations research to nonmonotonic reasoning.

(Nerode et al., 1995) are the first to propose a method of computing circumscription using integer programming. They claimed their method was correct for circumscription with fixed predicate, but we show that their method does not correctly reflect their claim. We show a correct method of computing all the minimal models not only with fixed predicates but also with varied predicates and we extend our method to compute prioritized circumscription as well.

Introduction
In this paper, we discuss a method of computing circumscription using integer programming used in operations research. Circumscription (McCarthy, 1980) has been proposed as a formalization of nonmonotonic reasoning and has been intensively studied. However, like other formalisms of nonmonotonic reasoning, it has a high complexity of computation and many proposals are made (Lifschitz, 1983; Przymusinski, 1989; Ginsberg, 1989; Nerode et al., 1995).

This paper gives a condition in which circumscription is collapsed into the first-order logic. (Ginsberg, 1989) and (Przymusinski, 1989) give methods which use theorem prover techniques. (Bell et al., 1992; Bell et al., 1996) and (Nerode et al., 1995) take different approaches from the above approaches. Circumscription is restricted to a propositional logic or a first-order sentences with domain closure axioms and unique name axioms. Then, they translate axioms into inequality constraints in integer programming and use a minimization of an objective function which corresponds with minimized predicates and obtain all the minimal models. This kind of research is very important since it introduces an usage of efficient method developed in operations research to nonmonotonic reasoning.

In this paper, we give a computing method of circumscription using integer programming. Circumscription (McCarthy, 1986) has been proposed as a formalization of nonmonotonic reasoning and intensively studied. However, it has a high complexity of computation and many proposals are made (Lifschitz, 1985; Przymusinski, 1989; Nerode et al., 1995).

We represent an interpretation as a set of true propositions in the interpretation.

In circumscription, there are three kinds of predicates: minimized predicates, fixed predicates, and varied predicates. Minimized predicates are subject to minimization whereas interpretation of fixed predicates cannot be changed for minimization, but interpretation of varied predicates can be changed if their change leads to further minimization of minimized predicates. (Bell et al., 1992; Bell et al., 1996) consider minimization of all the predicates and (Nerode et al., 1995) claim that they extend the method of (Bell et al., 1992; Bell et al., 1996) so that their method is correct for circumscription even including fixed predicates (but not including varied predicates). However, we show that their claim is not correct.

Even if their claim were correct, circumscription without varied predicates would have a serious drawback to apply circumscription to commonsense reasoning as Etherington et al. (Etherington et al., 1983) have pointed out.

For example, consider the following axioms.

\[\text{bird} \land \neg ab \supset \text{fly}.\]
\[\text{bird}.\]

It seems that circumscribing \(ab\) would yield \(\text{fly}\). However, without \(\text{fly}\) varied, it is impossible to derive \(\text{fly}\). This is because in this circumscription without \(\text{fly}\) varied, the interpretations are not comparable to each other if the interpretations of \(\text{fly}\) are different. There are three models of the above axioms, \(I_1 = \{\text{bird, ab, fly}\}\), \(I_2 = \{\text{bird, fly}\}\), \(I_3 = \{\text{bird, ab}\}\).

In minimizing \(ab\) without \(\text{fly}\) varied, \(I_2 < I_1\) holds, but \(I_2 < I_3\) does not hold since the interpretation of \(\text{fly}\) in \(I_2\) is different from the interpretation of \(\text{fly}\) in \(I_3\). So, minimal models for this circumscription are \(I_2\) and \(I_3\), and therefore, we cannot conclude \(\text{fly}\).

If we let the interpretation of \(\text{fly}\) be varied, then \(I_2\) is the only minimal model and therefore, we can conclude \(\text{fly}\). Therefore, usage of varied propositions is very important in commonsense reasoning.

In this paper, we give a computing method of circumscription for a propositional logic or a first-order logic with domain closure axioms and unique name axioms.

\(^1\text{We represent an interpretation as a set of true propositions in the interpretation.}\)
Our method can compute minimal models for this class of axioms not only with fixed predicates, but also with varied predicates. Moreover, [Nerode et al., 1993] gives a checking method of circumscriptive entailment for a limited class of formulas, whereas we give a complete checking method. Then, we extend our method to apply for prioritized circumscription as well. [Cadoli et al., 1992] propose a method of eliminating varied predicates in circumscription by translating inference problem of a formula under circumscription with varied predicates and fixed predicates into another inference problem under circumscription without varied predicates nor fixed predicates. So, readers might think that methods of [Bell et al., 1992; Bell et al., 1996] which compute all the minimal models without varied nor fixed propositions are suitable for computing minimal models. However, it is not clear how to apply the method proposed by [Cadoli et al., 1992] to computing minimal models since the relationship between a model of the original circumscription and a model of the translated circumscription is not known.

Preliminaries

We restrict our attention to propositional circumscription. For the first-order case with domain closure axioms and unique name axioms, we can translate each ground atom into a set of clauses of the form $\bigvee_i p_i \vee \neg q_i$. We associate each propositional symbol $p$ with variable $X_p$ for 0-1 variable which represents the truth value of $p$; If $X_p = 1$, $p$ is true and if $X_p = 0$, $p$ is false. We also use an interpretation $I$ to represent a solution of the assignments to variables from integer programming. If $p \in I$, it represents $X_p = 1$ and if $p \notin I$, it represents $X_p = 0$.

Let $F$ and $G$ be tuples of formulas, $(F_1, F_2, ..., F_n)$ and $(G_1, G_2, ..., G_n)$. We define $F \leq G$ as $\bigwedge_{i=1}^{n} F_i \supseteq G_i$. We define $F < G$ as $F \leq G$ and $G \not< F$, and $F \approx G$ as $F \leq G$ and $G \leq F$.

Let $A$ be a conjunction of formulas and $P$ be a set of propositional symbols used in $A$. We divide $P$ into disjoint three tuples of propositions $P, Z, Q$ which are called minimized propositions, varied propositions, and fixed propositions.

Circumscription of $P$ for $A$ with $Z$ varied is defined as follows.

$$ Circum(A; P; Z) = A(P; Z) \land \neg \exists p \exists z (A(p; z) \land p < P). $$

For a model theory of circumscription, we define an order of interpretations to minimize $P$ with $Z$ varied as defined as follows. Let $I$ be an interpretation and $\Phi$ be a tuple of propositional symbols. We define $[I[\Phi]$ as $\{ p \in \Phi | I(p) = p \}$ or, equivalently, $I \cap \Phi$.

Let $I_1$ and $I_2$ be interpretations.

$I_1 \leq^{P:Z} I_2$ if

**Step 1:** Let $AC := \emptyset$ and $SS := \emptyset$.

**Step 2:** Minimizing $\sum_{p \in P} X_p$ under $Tr(A) \cup AC$ using 0-1 integer programming.

**Step 3:** If there is no solution for the above minimization, output $SS$.

**Step 4:** Otherwise,

1. Let $M$ be a solution of the above minimization.
2. Add $M[P]$ to $SS$.
3. Add $\sum_{p \in M[P]} X_p \leq |M[P]| - 1$ to $AC$.
4. Go to Step 2.

![Figure 1: The algorithm of Nerode et al.](image)

1. $I_1[Q] = I_2[Q]$.
2. $I_1[P] \subseteq I_2[P]$.

We define $I_1 <^{P:Z} I_2$ as $I_1 \leq^{P:Z} I_2$ and $I_2 \not<^{P:Z} I_1$. A minimal model $M$ of $A(P, Z)$ w.r.t. $P$ with $Z$ varied is defined as follows.

1. $M$ is a model of $A(P, Z)$.
2. There is no model $M'$ of $A(P, Z)$ such that $M' <^{P:Z} M$.

According to [Lifschitz, 1985], $I$ is a minimal model of $A(P, Z)$ w.r.t. $P$ with $Z$ varied if and only if $I$ is a model of $Circum(A; P; Z)$.

Computing Minimal Models without Varied Propositions

Let $A$ be a set of clauses. Then, a set of inequalities, $Tr(A)$, translated from $A$ is defined as follows.

$$ Tr(A) = \{ X_{p_1} + ... + X_{p_m} + (1 - X_{q_1}) + ... + (1 - X_{q_n}) \geq 1 | p_1 \lor ... \lor p_m \lor \neg q_1 \lor ... \lor \neg q_n \in A \} $$

Let $Z$ be empty. Then, the algorithm proposed in [Nerode et al., 1993] in Figure 1. We adapt their algorithm for propositional circumscription. The algorithm works as follows. We start with $Tr(A)$ as the initial constraints and minimize an objective function corresponding with minimized propositions under $Tr(A)$. If we do not obtain any solution, we are done. Otherwise, we add a constraint $AC$ which excludes non-minimal models larger than the obtained solution.

[Nerode et al., 1993] claims the following on the correctness and completeness of the above algorithm.

**Claim:** [Nerode et al., 1993] (Theorem 1) Output $SS$ from the algorithm in Figure 1 is equivalent to $\{ M[P] | M$ is a minimal model of $A(P)$ with respect to $P$ with no propositions varied $\}$.

Unfortunately, this claim is not correct in general as the following example shows.
Step 1: Let $AC := \emptyset$ and $SS := \emptyset$.

Step 2: Minimizing $\sum_{p \in P} X_p$ under $Tr(A) \cup AC$ using 0-1 integer programming.

Step 3: If there is no solution for the above minimization, output $SS$.

Step 4: Otherwise,
1. Let $M$ be a solution of the above minimization.
2. Add $M$ to $SS$.
3. Add $\sum_{p \in M[P]} X_p + \sum_{q \in M[Q]} X_q \geq 1 - \sum_{q \in M[Q]} X_q \leq |M[P]| + |Q| - 1$ to $AC$.
4. Go to Step 2.

Figure 2: Algorithm for circumscription with fixed propositions

Example 1 Let $A(ab)$ be the following set of clauses.

$\neg bird \lor ab \lor fly$

$bird.$

Then, the minimal models of $A(ab)$ with respect to $\langle ab \rangle$ are $M_1 = \{bird, fly\}$ and $M_2 = \{bird, ab\}$. Note that fly is a fixed proposition and so, the two models are incomparable since interpretations of fly are different in these two models.

However, from the algorithm in Figure 2, we cannot obtain $M_2$ as follows.

$Tr(A)$ is

$1 - X_{bird} + X_{ab} + X_{fly} \geq 1$

$X_{bird} \geq 1$

By minimizing $X_{ab}$ using 0-1 integer programming under $Tr(A)$, we obtain a solution $X_{ab} = 0, X_{bird} = 1, X_{fly} = 1$ which corresponds with $M_1$.

Then, we add $M_1[\langle ab \rangle] = \emptyset$ to $SS$ and we add the following constraint to $AC$.

$0 \leq -1$.

Obviously, we cannot get any further solution. This means that we cannot obtain a minimal model $M_2$.

Therefore, the above claim does not work in general if there is a fixed proposition. Although their method is not correct with circumscription with fixed propositions, we later show that their method actually works for circumscription with varied propositions without fixed propositions.

Now, we give an algorithm which works correctly for circumscription with fixed propositions in Figure 2. Let $I$ be an interpretation and $\Phi$ be a tuple of propositional symbols. We define $\mathcal{T}(\Phi)$ used in Figure 2 as $\{p \in \Phi | I \neq p\}$ or equivalently, $\Phi - I$.

Theorem 1 Output $SS$ from the algorithm in Figure 2 is equivalent to $\{M | M$ is a minimal model of $A(P)$ with respect to $P$ with no propositions varied $\}$.

Proof: Let $\alpha$ be a formula which consists of logical connectives and propositional symbols in $P$. Then, according to (Le Kleer and Konolige, 1985), Circum$(A; P) \models \alpha$ if and only if Circum$(A \land (R \equiv \neg \cdot Q); P, Q, R) \models \alpha$ where $R$ is a tuple of new propositions not in $A$ and $\neg \cdot Q$ is $\langle\neg q_1, \ldots, \neg q_m\rangle$ for $Q = \langle q_1, \ldots, q_m\rangle$. Then, we use the algorithm of (Bell et al., 1992) to minimize all propositions and replace every occurrence of variables $X_r$ for a proposition $r_i$ in $R$ by $1 - X_{q_i}$. $\square$

Example 2 Let $A(ab)$ be the following set of clauses as in Example 1

$\neg bird \lor ab \lor fly$

$bird.$

Then, the minimal models of $A(ab)$ with respect to $\langle ab \rangle$ are $M_1 = \{bird, fly\}$ and $M_2 = \{bird, ab\}$.

By minimizing $X_{ab}$ under $Tr(A)$, we obtain a solution $X_{ab} = 0, X_{bird} = 1, X_{fly} = 1$ which corresponds with a minimal model $M_1$.

Then, we add $M_1$ to $SS$ and we add the following constraint to $AC$.

$X_{bird} + X_{fly} \leq 1$.

Then, minimizing $X_{ab}$ under $Tr(A) \cup AC$, we obtain a solution $X_{ab} = 0, X_{bird} = 1, X_{fly} = 0$ which corresponds with a minimal model $M_2$.

Then, we add $M_2$ to $SS$ and we add the following constraint to $AC$.

$X_{ab} + X_{bird} + (1 - X_{fly}) \leq 2$

Then, minimizing $X_{ab}$ under $Tr(A) \cup AC$, we no longer obtain any solution and therefore, $SS = \{\{bird, fly\}, \{bird, ab\}\}$ is obtained.

Computing Minimal Models with Varied Propositions

As shown in Introduction, we need varied proposition to perform commonsense reasoning. We give a computation method of handling varied propositions in Figure 3.

Let $F, G$ be disjoint sets of propositions. We define $\mathcal{F}(F, G)$ as

$\bigwedge_{f \in F} f \land \bigwedge_{f \in G} \neg f$.

Theorem 2 Output $MS$ from the algorithm in Figure 3 is equivalent to $\{M | M$ is a minimal model of $A(P, Z)$ with respect to $P$ with $Z$ varied $\}$.

Example 3 Let $A(ab)$ be the following set of clauses.

$\neg bird \lor ab \lor fly$

$bird.$

Then, the minimal model of $A(ab)$ with respect to $\langle ab \rangle$ with $\langle fly \rangle$ varied is $M_1 = \{bird, ab\}$.

By minimizing $X_{ab}$ under $Tr(A)$, we obtain a solution where $X_{ab} = 0, X_{bird} = 1$ and $X_{fly} = 1$. We add
Computing Minimal Models in Prioritized Circumscription

We firstly give a definition of prioritized circumscription. We divide a set of propositions into \( n \) partitions and give an order over partitions. Suppose that this is \( P_1 < P_2 < \ldots < P_n \). Intended meaning of this order is that we firstly minimize \( P_1 \), then \( P_2 \), ..., then \( P_n \). Let \( P \) and \( Q \) be a tuple of propositions which have orders \( P_1 < P_2 < \ldots < P_n \) and \( Q_1 < Q_2 < \ldots < Q_n \). We define \( P \preceq Q \) as follows. If \( i = 1 \), \( P \preceq Q \) is \( P_1 \leq Q_1 \) and if \( i > 1 \), \( (\bigwedge_{j=1}^{n-1} P_j \approx Q_j) \cup P_i \leq Q_i \). We define \( P \preceq Q \) as \( \bigwedge_{i=1}^{n} P_i \preceq Q_i \) and \( P < Q \) as \( P \preceq Q \) and \( Q \nleq P \).

Prioritized circumscription of \( P_1 < P_2 < \ldots < P_n \) for \( A \) with \( Z \) varied is defined as follows.

\[
\text{Circum}(A; P_1 < P_2 < \ldots < P_n; Z) = A(P; Z) \land \neg \exists p \exists z(A(p, z) \land p < P).
\]

In a model theory of prioritized circumscription, we define an order over interpretations as follows.

Let \( I_1 \) and \( I_2 \) be interpretations and let \( P \) consist of disjoint sets \( P_1, P_2, \ldots, P_n, Q, Z \).

\[
I_1 \preceq I_2 \text{ if and only if } P_1 < P_2 < \ldots < P_n; Z \preceq I_2 \text{ and } I_1 \nleq I_2; Z I_1.
\]

A minimal model \( M \) of \( A(P; Z) \) w.r.t. \( P_1 < P_2 < \ldots < P_n; Z \) with \( Z \) varied is defined as follows.

1. \( M \) is a model of \( A(P; Z) \).
2. There is no model \( M' \) of \( A(P; Z) \) such that \( M' \preceq P_1 < P_2 < \ldots < P_n; Z \preceq M \).

According to [Lifschitz, 1983], \( I \) is a minimal model of \( A(P; Z) \) w.r.t. \( P_1 < P_2 < \ldots < P_n; Z \) with \( Z \) varied if \( I \) is a model of \( \text{Circum}(A; P_1 < P_2 < \ldots < P_n; Z) \).

Similar to the problem in non-prioritized circumscription, the method proposed in [Nerode et al., 1993] of computing prioritized circumscription is correct if there are no fixed propositions.

To manipulate fixed propositions in prioritized circumscription, we need the following theorem which is a generalization of the result of [de Kleer and Konolige, 1989].

**Theorem 3** Let a set of propositions \( P \) consist of disjoint sets \( P_1, P_2, \ldots, P_n, Q, Z \) and \( P = P_1 \cup P_2 \cup \ldots \cup P_n \) and \( \alpha \) be a formula which consists of logical connectives and propositional symbols in \( P \). Then,

\[
\text{Circum}(A(P; Z); P_1 > P_2 > \ldots > P_n; Z) \models \alpha \text{ if and only if } \text{Circum}(A(P; Z) \land (R \equiv \neg \cdot Q); Q, R, P_1 > P_2 > \ldots > P_n; Z) \models \alpha.
\]

This theorem means that we can translate prioritized circumscription with fixed propositions to prioritized circumscription without fixed propositions. Moreover, we extend the method so that it is applicable even if there are varied propositions. We show the algorithm in Figure 2.
Step 1: Let $AC := \emptyset$ and $SS := \emptyset$.

Step 2: Minimizing $\sum_{p \in P_i} X_p$ under $Tr(A) \cup AC$ using 0-1 integer programming.

Step 3: If there is no solution for the above minimization, go to Step 5.

Step 4: Otherwise,

1. Let $M$ be a solution of the above minimization.
2. Add $M[\{P_1 \cup Q\}]$ to $SS$
3. Add $\sum_{p \in M[\{P_1\}]} X_p + \sum_{q \in M[\{Q\}]} X_q + \sum_{q' \in M[\{Q\}]} (1 - X_{q'}) \leq |M[\{P_1\}]| + |\{Q\}| - 1$ to $AC$.
4. Go to Step 2.

Step 5: For $i := 2$ to $n$ do the following.

1. $SS' := \emptyset$.
2. For every $S \in SS$ do
   
   Step 5-1: Let $AC := \emptyset$.

   Step 5-2: Minimizing $\sum_{p \in \bigcup_{i=1}^{n} P_i} X_p$ under $Tr(A \land F(S, (P_1 \cup \ldots \cup P_{i-1} \cup Q) - S)) \cup AC$ using 0-1 integer programming.

   Step 5-3: If there is no solution for the above minimization, process the next $S$.

   Step 5-4: Otherwise,
   
   (a) Let $M$ be a solution of the above minimization.
   (b) Add $M[\{P_1 \cup \ldots \cup P_i \cup Q\}]$ to $SS'$.
   (c) Add $\sum_{p \in M[\{P_i\}]} X_p \leq |M[\{P_i\}]| - 1$ to $AC$.
   (d) Go to Step 5-2.
3. $SS := SS'$ and do the next iteration for $i$.

If iteration stops then let $MS$ be $\emptyset$ and for every $S \in SS$ do the following.
1. Let $A' := A \land F(S, (P_i \cup \ldots \cup P_n \cup Q) - S)$.
2. Compute all the models of $A'$ and add these models to $MS$.
Output $MS$.

Figure 4: Algorithm for prioritized circumscription

**Theorem 4** Output $MS$ from the algorithm in Figure 3 is equivalent to 

$\{M | M$ is a minimal model of $A(P, Z)$ with respect to $P_1 < \ldots < P_n$ with $Z$ varied$\}$.

**Example 4** Consider the following axioms.

\begin{align*}
&ab \lor \neg f ly \\
&\neg bird \lor ab_2 \lor f ly.
\end{align*}

We compute minimal models of $\text{Circum(A)}(\{ab_2\} > \{ab_1\}; \{f ly\})$ meaning that we minimize $ab_1$ and $ab_2$ with $f ly$ varied (and $bird$ fixed) and $ab_2$ is preferably minimized than $ab_1$. The minimal models are $\{bird, f ly, ab_1\}$ and $\emptyset$. We have two minimal models since the interpretations of $bird$ in these models are different from each other.

Step 1: $AC := \emptyset$ and $SS := \emptyset$.

Step 2(1): Minimize $X_{ab_2}$ under the following constraints:

\begin{align*}
&X_{ab_2} + 1 - X_{f ly} \geq 1 \\
&1 - X_{bird} + X_{ab_2} + X_{f ly} \geq 1
\end{align*}

Step 3(1): Then, there are three solutions for this minimization:

\begin{align*}
&S_1 = \{X_{ab_2} = 0, X_{bird} = 1, X_{f ly} = 1, X_{ab_1} = 1\}, \\
&S_2 = \{X_{ab_2} = 0, X_{bird} = 0, X_{f ly} = 1, X_{ab_1} = 1\}, \\
&S_3 = \{X_{ab_2} = 0, X_{bird} = 0, X_{f ly} = 0, X_{ab_1} = 0\}.
\end{align*}

Step 4(1): Suppose that we obtain $S_1$.

1. $M_1 = \{\text{bird}, f ly, ab_1\}$
2. Add $M_1[\{ab_2 \cup \{bird\}\}] = \{\text{bird}\}$ to $SS$. ($SS$ becomes $\{\{\text{bird}\}\}$.)
3. Add $X_{bird} \leq 0$ to $AC$.

Step 2(2): Minimize $X_{ab_2}$ under new $AC$.

Step 3(2): Then, there are two solutions for this minimization, $S_2$ and $S_3$.

Step 4(2): Suppose that we obtain $S_2$.

1. $M_2 = \{f ly, ab_1\}$
2. Add $M_2[\{ab_2 \cup \{bird\}\}] = \emptyset$ to $SS$. ($SS$ becomes $\{\{\text{bird}\}, \emptyset\}$.)
3. Add $1 - X_{bird} \leq 0$ to $AC$.

Step 2(3): Minimize $X_{ab_1}$ under new $AC$.

Step 3(3): Then, we no longer obtain any solutions, and go to Step 5.

Step 5:

$i := 2$ and $SS' := \emptyset$.

1. $S := \{\text{bird}\}$.

Step 5-1(1): $AC := \emptyset$.

Step 5-2(1): Minimize $X_{ab_1}$ under the following constraints:

\begin{align*}
&X_{ab_1} + 1 - X_{f ly} \geq 1 \\
&1 - X_{bird} + X_{ab_2} + X_{f ly} \geq 1 \\
&X_{ab_2} \leq 0 \\
&X_{bird} \geq 1
\end{align*}

Step 5-3(1): Then, we obtain the solution $S_1$ again.

Step 5-4(1):

(a) $M_1 = \{\text{bird}, f ly, ab_1\}$
(b) Add $M_1[\langle ab_0 \rangle \cup \langle ab_1 \rangle \cup \langle \text{bird} \rangle] = \{ab_1, \text{bird}\}$ to $SS'$. ($SS'$ becomes $\{\{ab_1, \text{bird}\}\}$.)

(c) Add $X_{ab_1} \leq 0$ to $AC$.

**Step 5-2(2):** Minimize $X_{ab_1}$ under new $AC$.

**Step 5-3(2):** Then, we no longer obtain any solutions.

2. $S := \emptyset$.

**Step 5-1(1):** $AC := \emptyset$.

**Step 5-2(1):** Minimize $X_{ab_1}$ under the following constraints:

\[
\begin{align*}
X_{ab_1} + 1 - X_{\text{fly}} &\geq 1 \\
1 - X_{\text{bird}} + X_{ab_2} + X_{\text{fly}} &\geq 1 \\
X_{ab_2} &\leq 0 \\
X_{\text{bird}} &\leq 0
\end{align*}
\]

**Step 5-3(1):** Then, we obtain the solution $S_3$ only.

**Step 5-4(1):**

(a) $M_3 = 0$

(b) Add $M_3[\langle ab_2 \rangle \cup \langle ab_1 \rangle \cup \langle \text{bird} \rangle] = \emptyset$ to $SS'$. ($SS'$ becomes $\emptyset$.)

(c) Add $0 \leq -1$ to $AC$.

**Step 5-2(2):** Minimize $X_{ab_1}$ under new $AC$.

**Step 5-3(2):** Then, we no longer obtain any solutions.

Iteration stops and by calculation of $MS$ from $SS'$, we obtain $\{\{\text{bird, fly, } ab_1\}, \emptyset\}$. We can also give a method of circumscriptive entailment in prioritized circumscription as in ordinary circumscription. After iteration stops, we check for every $A' \in SS$, $A' \land \lnot \alpha$ does not have any models to check whether $\alpha$ is consequence of the prioritized circumscription or not.

**Conclusion**

Contributions of this paper are as follows.

1. We correctly give the method of computing all the models of circumscription not only with fixed propositions, but also with varied propositions.

2. We give a complete method of computing circumscriptive entailment for propositional logic.

3. We also extend the method of computing minimal models to include varied propositions in prioritized circumscription.

**Acknowledgements** This research is partly supported by Grant-in-Aid for Scientific Research (Project No. 11878067), The Ministry of Education, Japan. We also thank the anonymous referees for comments on this paper.

**References**

(Bell et al., 1996) Bell, C., Nerode, A., Ng, R., and Subrahmanian, V. S. 1996. Implementing Deductive Database by Mixed Integer Programming. *ACM Transactions on Database Systems*, 21, pages 238 – 269.

(Cadoli et al., 1992) Cadoli, M., Eiter, T., and Gottlob, G. 1992 An Efficient Method for Eliminating Varying Predicates from a Circumscription. *Artificial Intelligence*, 54, pages 397 – 410.

(de Kleer and Konolige, 1989) de Kleer, J., and Konolige, K. 1989. Eliminating the Fixed Predicates from a Circumscription. *Artificial Intelligence*, 39, pages 391 – 398.

(Etherington et al., 1985) Etherington, D., Mercer, R., and Reiter, R. 1985. On the Adequacy of Predicate Circumscription for Closed-world Reasoning. *Computational Intelligence*, 1, pages 11 – 15.

(Ginsberg, 1989) Ginsberg, M. 1989. Circumscriptive Theorem Prover. *Artificial Intelligence*, 39, pages 209 – 230.

(Lifschitz, 1985) Lifschitz, V. 1985. Computing Circumscription. *Proc. of IJCAI-85*, pages 121 – 127.

(McCarthy, 1986) McCarthy, J. 1986. Applications of Circumscription to Formalizing Common-Sense Knowledge. *Artificial Intelligence*, 28, pages 89 – 116.

(Nerode et al., 1995) Nerode, A., Ng, R. T., and Subrahmanian, V. S. 1995. Computing Circumscriptive Databases, I. Theory and Algorithms. *Information and Computation*, 116, pages 58–80.

(Przymusinski, 1989) Przymusinski, T. 1989. An algorithm to compute circumscription. *Artificial Intelligence*, 38, pages 49 – 73.