Chapter 4
Compiling Circumscription into ELP

4.1 Introduction

Gelfond and Lifschitz [1988a] were the first to explore the compiling approach to compute circumscription based on the relationship between the semantics of circumscription and the semantics of logic programs. They presented a method of compiling some restricted class of prioritized circumscription into a stratified logic program. Though their method is computationally efficient, the applicable class is too limited.

So, as discussed in Chapter 3, we proposed an extension of Gelfond and Lifschitz’s method which also compile prioritized circumscription into a stratified logic program [Wakaki and Satoh, 1995]. With keeping the efficiency of Gelfond and Lifschitz’s method, this method expands the applicable class of circumscription by making use of the Lifschitz’s result [Lifschitz, 1985] about parallel circumscription of a solitary formula. However, as far as a class of stratified logic programs is considered as the target language to which circumscription is compiled, the applicable class is limited within a class of circumscription which has a unique minimal model since every stratified logic program has a unique perfect model [Przymusinski, 1987]. But there are many examples of nonmonotonic reasoning whose intended meaning cannot be represented by a unique model such as multiple extension problem, a class of circumscription which has fixed predicates, and so on.

Recently Sakama and Inoue [1995; 1996] proposed two methods both of which compile circumscription into classes of more general logic programs whose semantics are given by stable models for the first one [Sakama and Inoue, 1995] and by preferred answer sets for the second one [Sakama and Inoue, 1996]. Though both of their methods can handle the multiple extension problem as well as circumscription with fixed predicates, the first one is only applicable to parallel circumscription but not to prioritized circumscription. On the other hand, the second one is applicable to prioritized circumscription, but it gives only the semantic aspects and is lack of the feasible logic programming interpreter for prioritized logic programs proposed by them as the target language into which prioritized circumscription is
In this chapter, we present a new method of compiling circumscription into extended logic programs proposed by Gelfond and Lifschitz [1991] as its target language. It is widely applicable to a class of parallel circumscription as well as a class of prioritized circumscription. Showing the semantic correspondence between circumscription with fixed predicates and Reiter’s default theory which generalizes Theorem 2 in [Etherington, 1987a] (see Theorem 2.18 in chapter 2), we can give not only the semantic relationship between a class of parallel circumscription and a class of extended logic programs but also the one between a class of prioritized circumscription and a class of extended logic programs. As a result, circumscription whose theory contains both the domain closure axiom and the uniqueness of names axioms and does not contain function symbols can always be compiled into an extended logic program $\Pi$, so that, whether a ground literal is provable from circumscription, can always be evaluated by deciding whether the literal is true in all answer sets of $\Pi$.* This can be computed by running $\Pi$ under the existing logic programming interpreter such as Satoh and Iwayama’s top-down query evaluation procedure for abductive logic programming [Satoh and Iwayama, 1992].

Finally, we present that our approach exploiting classical negation $\neg$ can also give an extension of Sakama and Inoue’s first method [1995] to make it possible to compute prioritized circumscription.

This chapter is organized as follows. In Section 4.2, we provide two syntactical definitions of extended logic programs into which parallel circumscription and prioritized circumscription are compiled respectively. Then, we present theorems and corollaries on which our method is theoretically based, along with examples. In Section 4.3, we compare our method with related researches. We finish Section 4.4 with giving proofs of theorems in this chapter.

### 4.2 Translation from Circumscription to ELP

We give a translation from circumscription to extended logic programs. Parallel circumscription is translated into ELP $\Pi^\alpha$ and prioritized circumscription is translated into ELP $\Pi^\beta$ respectively.

Firstly, we show the definition of characteristic clauses [Inoue, 1992a] which are needed in our translation.

**Definition 4.1** [Inoue, 1992a] Let $\Sigma$ be a set of clauses. Then $Th(\Sigma)$ stands for a set of clauses which are theorems of $\Sigma$. A clause in $Th(\Sigma)$ which is not properly subsumed by any theorem in $Th(\Sigma)$ is called a characteristic clause. $\mu Th(\Sigma)$ denotes the set of all characteristic clauses in $Th(\Sigma)$.

In the following Definition 4.2 and Definition 4.3, we restrict a first order theory $T$ to the one which is function-free and contains both the domain closure axiom

* In this paper, we call this method “Wakaki and Satoh’s method.”
We consider $T$ as a set of clauses as follows:

$$T \overset{\text{def}}{=} \Sigma \cup \text{DCA} \cup \text{UNA}. \quad (4.1)$$

Let $\Sigma^{\text{DCA}}$ denote a set of ground clauses such that every clause in $\Sigma$ containing variables is replaced by all its ground instances obtained by substituting every ground term in DCA for each variable. $t$ stands for a tuple of ground terms occurring in DCA.

**Definition 4.2** Given parallel circumscription $\text{Circum}(\exists\forall T; P; Z)$ where $\exists\forall$ is a universal closure, ELP $\Pi^\alpha$ is constructed as follows:

1. For any minimized predicate $p$ from $P$ and any $t$, $\Pi^\alpha$ has a rule:

$$\neg p(t) \leftarrow \text{not } p(t).$$

2. For any fixed predicate $q$ from $Q$ and any $t$, $\Pi^\alpha$ has two rules as follows:

$$\neg q(t) \leftarrow \text{not } q(t),$$

$$q(t) \leftarrow \text{not } \neg q(t).$$

3. For any characteristic clause $C \overset{\text{def}}{=} \ell_1 \lor \ell_2 \lor \ldots \lor \ell_n$ in $\mu \text{Th}(\Sigma^{\text{DCA}})$ and any contrapositive of $C$ such as:

$$\neg \ell_1 \land \ldots \land \neg \ell_{i-1} \land \neg \ell_{i+1} \land \ldots \land \neg \ell_n \supset \ell_i \quad (1 \leq i \leq n),$$

$\Pi^\alpha$ has rules as follows:

$$\ell_i \leftarrow \neg \ell_1 \land \ldots \land \neg \ell_{i-1} \land \neg \ell_{i+1} \land \ldots \land \neg \ell_n.$$

**Definition 4.3** Given prioritized circumscription as follows:

$$\text{Circum}(\exists\forall T(P^1 \ldots P^k, Z, Q); P^1 > P^2 > \ldots > P^k; Z),$$

where $P^r (1 \leq r \leq k), Z$ and $Q$ are tuples of predicate symbols such as $[(P^r)_1, \ldots, (P^r)_{\ell_r}], [Z_1, \ldots, Z_m]$ and $[Q_1, \ldots, Q_n]$. ELP $\Pi^\beta$ is constructed in the following two steps:

1. According to Theorem 2.10, a given prioritized circumscription is represented by the conjunction of $k$ parallel circumscriptions. So, let every $ith (1 \leq i \leq k)$ parallel circumscription,

$$\text{Circum}(\exists\forall T(P^1 \ldots P^k, Z, Q); P^1; P^{i+1}, \ldots, P^k; Z)$$

be transformed in such a way that all predicate symbols occurring in it are renamed by using $P^i_1, \ldots, P^i_k, Z_i, Q_i$ instead of $P^1, \ldots, P^k, Z, Q$, which leads to as follows:

$$\text{Circum}(\exists\forall T_i; P^i_1; P^{i+1}, \ldots, P^k, Z_i), \quad (4.2)$$

where $T_i$ denotes $T(P^i_1, \ldots, P^i_k, Z_i, Q_i)$, and $P^{i_r}, Z_i, Q_i$ are tuples of predicate symbols such as $[(P^{i_r})_1, \ldots, (P^{i_r})_{\ell_r}],[Z_i)_1, \ldots, (Z_i)_{m}], [(Q_i)_1, \ldots, (Q_i)_n]$. 
2. ELP $\Pi^\beta$ consists of all rules from $(\Pi^\alpha)^1, \ldots, (\Pi^\alpha)^k$ and $\Pi_\gamma$ where

(a) each $(\Pi^\alpha)^i$ is an extended logic program (ELP) which is constructed from the $i$th renamed parallel circumscription (4.1) according to Definition 4.2.

(b) $\Pi_\gamma$ is an extended logic program which consists of the following rules:

For any predicate $u$ from $P, Z, Q$ and any $t$,

\[
\begin{align*}
u(t) & \leftarrow u_i(t), \\
\neg u(t) & \leftarrow \neg u_i(t). \quad (1 \leq i \leq k)
\end{align*}
\]

where $u$ and $u_i$ stand for any predicate symbol of $(Pr)^f, (Zg)_g, (Qh)_h$ and any one of $(Pr)^f, (Zg), (Qh)$ respectively.

$(1 \leq f \leq \ell_r, 1 \leq g \leq m, 1 \leq h \leq n)$

First of all, we show the following theorem which generalizes Theorem 2.18 (i.e. Theorem 2 in [Etherington, 1987a]) based on Theorem 1 in [de Kleer and Konolige, 1989].

**Theorem 4.4** Assume that $T$ is a first order theory including $DCA$ and $UNA$. Let $P, Q$ and $Z$ be a tuple of minimized predicates, a tuple of fixed predicates and a tuple of variable predicates respectively. Then, for any first-order formula $F$ of the language of $T$, it holds that,

\[
\text{Circum}(T; P, Z) \models F \iff \text{for any extension } E \text{ of a default theory } \Delta, E \models F
\]

where $\Delta$ is defined as follows:

\[
\Delta \overset{\text{def}}{=} \left\{ \frac{-p(x)}{-p(x)}, \frac{-q(x)}{-q(x)}, \frac{q(x)}{q(x)} \mid p \in P, q \in Q \right\}, T
\]

Based on Theorem 4.4 as well as a 1-1 correspondence between the extensions of a default theory and the answer sets of an extended logic program shown in [Gelfond and Lifschitz, 1991], the semantic relationships between circumscription and the translated ELPs $\Pi^\alpha, \Pi^\beta$ are given as follows.

**Theorem 4.5** For any ground literal $G$ of the language of $T$ whose predicate symbol is not equality, it holds that,

\[
\text{Circum}(\forall T; P, Z) \models G \iff G \text{ is true in all answer sets of ELP } \Pi^\alpha,
\]

where ELP $\Pi^\alpha$ is constructed from $\text{Circum}(\forall T; P, Z)$ by using definition 4.2.

**Theorem 4.6** For any ground literal $G$ of the language of $T$ whose predicate symbol is not equality, it holds that,
4.2 TRANSLATION FROM CIRCUMSCRIPTION TO ELP

\[ \text{Circum}(\forall T; P^1 > P^2 > \ldots > P^k; Z) \models G \]
\[ \text{iff } G \text{ is true in all answer sets of ELP } \Pi^\beta. \]
where ELP \( \Pi^\beta \) is constructed from \( \text{Circum}(\forall T; P^1 > P^2 > \ldots > P^k; Z) \),
by using Definition 4.3.

**Remarks.** In this thesis, we refer to Theorem 4.5 and Theorem 4.6 as “Wakaki and Satoh’s method”.

Satoh and Iwayama [1992] show that whether a ground atom \( G \) is true in all stable models of a normal logic program \( \Pi \), can be decided by running their top-down query evaluation procedure for abductive logic programs where an abductive framework is given by \( \langle \Pi, A \rangle \) in which \( A \) is a set of predicate symbols called abducible predicates. Their result is given as follows:

Suppose that a normal logic program \( \Pi \) is consistent, which means that there exists a stable model of \( \Pi \), and all ground rules obtained by replacing all variables in each rule in \( \Pi \) by every element of its Herbrand universe are finite. Then it holds that,

\[ G \text{ is true in all stable models of } \Pi \]
\[ \text{iff } \text{derive}(\neg G, \{\}) \text{ fails} \]
under the abductive framework \( \langle \Pi, \{\} \rangle \).

where \( \text{derive} \) is a procedure given by them and \( \neg \) in its first argument denotes the negation-as-failure operator.

Since all ground rules in ELP \( \Pi^\alpha \) and \( \Pi^\beta \) constructed by using Definition 4.2 and Definition 4.3 respectively are finite, we can make use of their result in a slightly extended form, and Theorem 4.5 and Theorem 4.6 can be said in other words by the following corollaries.

**Corollary 4.7** Suppose that \( T \) is consistent and function-free. For any ground literal \( G \) of the language of \( T \) whose predicate symbol is not equality, it holds that,

\[ \text{Circum}(\forall T; P; Z) \models G \]
\[ \text{iff } \text{derive}(\neg G, \{\}) \text{ fails} \]
under the abductive framework \( \langle \Gamma^\alpha, \{\} \rangle \),
where \( \Gamma^\alpha \) consists of all rules of ELP \( \Pi^\alpha \) constructed from \( \text{Circum}(\forall T; P; Z) \) by using Definition 4.2 plus rules, \( \leftarrow u(t), \neg u(t) \), for every predicate \( u \) in \( T \).

**Corollary 4.8** Suppose that \( T \) is consistent and function-free. For any ground literal \( G \) of the language of \( T \) whose predicate symbol is not equality, it holds that,

\[ \text{Circum}(\forall T; P^1 > P^2 > \ldots > P^k; Z) \models G \]
\[ \text{iff } \text{derive}(\neg G, \{\}) \text{ fails} \]
under the abductive framework \( \langle \Gamma^\beta, \{\} \rangle \).
where \( \Gamma^\beta \) consists of all rules of ELP \( \Pi^\beta \) constructed from \( \text{Circum}(\forall T; P^1 > P^2 > \ldots > P^k; Z) \) by using Definition 4.3 plus rules, \( \leftarrow u(t), \neg u(t) \), for every predicate \( u \) in \( T \).

**Remarks.** There is Ginsberg’s early work [Ginsberg, 1989] on evaluating circumcription using abduction. But his method does not exploit the logic programming.
Example 4.9

Our compilation can handle circumscription representing *multiple inheritance*. Consider parallel circumscription:

$$\text{Circum}(\exists T; ab_1, ab_2; \text{pacifist})$$

(1)

where $$T \overset{\text{def}}{=} \Sigma \cup \text{DCA} \cup \text{UNA}$$. $$\Sigma$$ consists of the following clauses and DCA is $$x = \text{Nixon}$$:

$$\begin{align*}
\text{pacifist}(x) & \lor ab_1(x) \lor \neg \text{quaker}(x), \\
\neg \text{pacifist}(x) & \lor ab_2(x) \lor \neg \text{republican}(x), \\
\text{republican}(\text{Nixon}), & \quad \text{quaker}(\text{Nixon}).
\end{align*}$$

A set $$Q$$ of fixed predicates is $$\{\text{republican, quaker}\}$$ in this example. Hereafter we abbreviate $$\text{pacifist}, \text{republican}, \text{quaker}$$ and $$\text{Nixon}$$ to $$\text{pac}, \text{rep}, \text{quak}$$ and $$N$$ respectively.

After replacing a variable $$x$$ in every clause in $$\Sigma$$ by a ground term $$N$$ in DCA, $$\mu\text{Th}(\Sigma^{\text{DCA}})$$ is obtained as follows:

$$\begin{align*}
\text{pac}(N) & \lor ab_1(N), \\
\neg \text{pac}(N) & \lor ab_2(N), \\
ab_1(N) & \lor ab_2(N), \\
\text{rep}(N), & \quad \text{quak}(N).
\end{align*}$$

According to Definition 4.2, let us construct ELP $$\Pi^a$$ from circumscription (1).

1. For minimized predicates $$ab_1, ab_2$$ and a term $$N$$,

$$\begin{align*}
\neg ab_1(N) & \leftarrow \text{not } ab_1(N), \\
\neg ab_2(N) & \leftarrow \text{not } ab_2(N),
\end{align*}$$

2. For fixed predicates $$\text{rep, quak}$$ and a term $$N$$,

$$\begin{align*}
\neg \text{rep}(N) & \leftarrow \text{not } \text{rep}(N), \\
\text{rep}(N) & \leftarrow \text{not } \neg \text{rep}(N), \\
\neg \text{quak}(N) & \leftarrow \text{not } \text{quak}(N), \\
\text{quak}(N) & \leftarrow \text{not } \neg \text{quak}(N),
\end{align*}$$

3. For all contrapositives of all clauses of $$\mu\text{Th}(\Sigma^{\text{DCA}})$$,

$$\begin{align*}
ab_1(N) & \leftarrow \neg \text{pac}(N), \\
\text{pac}(N) & \leftarrow \neg ab_1(N), \\
ab_2(N) & \leftarrow \text{pac}(N), \\
\neg \text{pac}(N) & \leftarrow \neg ab_2(N), \\
ab_1(N) & \leftarrow \neg ab_2(N), \\
ab_2(N) & \leftarrow \neg ab_1(N), \\
\text{rep}(N) & \leftarrow, \quad \text{quak}(N) \leftarrow .
\end{align*}$$
As a result, \( \Pi^o \) has two answer sets:
\[
\{ \neg ab_1(N), ab_2(N), pac(N), rep(N), quak(N) \},
\{ ab_1(N), \neg ab_2(N), \neg pac(N), rep(N), quak(N) \}.
\]

**Example 4.10** Let us compile the following prioritized circumscription:
\[
\text{Circum}(\{ p \lor q, q \lor r \}; p > q; r)
\]
It holds that \( \Sigma = \mu Th(\Sigma) \) since \( \Sigma = \{ p \lor q, q \lor r \} \).
Contrapositives of \( p \lor q \) are as follows:
\[
\neg q \supset p, \quad \neg p \supset q,
\]
and those of \( q \lor r \) are as follows:
\[
\neg r \supset q, \quad \neg q \supset r.
\]
According to Theorem 2.10, it holds that
\[
\text{Circum}(\{ p \lor q, q \lor r \}; p > q; r)
\equiv \text{Circum}(\{ p \lor q, q \lor r \}; p, q, r)
\land \text{Circum}(\{ p \lor q, q \lor r \}; q, r).
\]
Then according to Definition 4.3, predicate symbols occurring each parallel circumscription are renamed as follows:
\[
\text{Circum}(\{ p_1 \lor q_1, q_1 \lor r_1 \}; p_1; q_1, r_1), \quad (2)
\text{Circum}(\{ p_2 \lor q_2, q_2 \lor r_2 \}; q_2; r_2). \quad (3)
\]
Therefore ELPs \((\Pi^o)_1, (\Pi^o)_2\) constructed from (2) and (3) respectively as well as \( \Pi_2 \) are obtained as follows:
\[
(\Pi^o)_1 : \begin{align*}
\neg p_1 & \leftarrow \text{not } p_1, \\
p_1 & \leftarrow \neg q_1, \\
q_1 & \leftarrow \neg p_1, \\
q_1 & \leftarrow \neg r_1, \\
r_1 & \leftarrow \neg q_1.
\end{align*}
\]
\[
(\Pi^o)_2 : \begin{align*}
\neg q_2 & \leftarrow \text{not } q_2, \\
\neg p_2 & \leftarrow \text{not } p_2, \\
p_2 & \leftarrow \text{not } \neg p_2, \\
p_2 & \leftarrow \neg q_2, \\
q_2 & \leftarrow \neg p_2, \\
q_2 & \leftarrow \neg r_2, \\
r_2 & \leftarrow \neg q_2.
\end{align*}
\]
\[ \Pi_\gamma: \quad p \leftarrow p_1, \quad p \leftarrow p_2, \\
q \leftarrow q_1, \quad q \leftarrow q_2, \\
r \leftarrow r_1, \quad r \leftarrow r_2, \\
\neg p \leftarrow \neg p_1, \quad \neg p \leftarrow \neg p_2, \\
\neg q \leftarrow \neg q_1, \quad \neg q \leftarrow \neg q_2, \\
\neg r \leftarrow \neg r_1, \quad \neg r \leftarrow \neg r_2. \]

ELP \( \Pi^\beta \) has the only one answer set:

\[ \{ \neg p, q, \neg p_1, q_1, \neg p_2, q_2 \}, \]

where \( \Pi^\beta \overset{\text{def}}{=} (\Pi^\alpha)_1 \cup (\Pi^\alpha)_2 \cup \Pi_\gamma. \)

Thus according to Theorem 4.6, we can conclude that

\[ \text{Circum}(\{ p \lor q, q \lor r \}; p > q; r) \models \neg p, \]
\[ \text{Circum}(\{ p \lor q, q \lor r \}; p > q; r) \models q, \]

but neither \( r \) nor \( \neg r \) is provable from this prioritized circumscription.

In the following, we give an extension of Sakama and Inoue’s first method [1995]. According to their method, parallel circumscription is translated into a general disjunctive program (GDP) whose semantics is given by stable models. Alternatively, they also show the translation of parallel circumscription into an Extended disjunctive program (EDP) whose semantics is given by the answer sets, which just corresponds to the stable models of the translated GDP. Each of their methods is applicable to a class of parallel circumscription, but not to prioritized circumscription. Our target language ELP has the expressive power of classical negation \( \neg \), which enables us to compute prioritized circumscription as is shown in Theorem 4.6. Since the classical negation is also available to EDP, we can make use of our method to extend their alternative method whose target language is EDP, so that it may become applicable to a class of prioritized circumscription as follows.

**Definition 4.11** [Sakama and Inoue, 1995]

Given parallel circumscription \( \text{Circum}(\forall T; P; Z) \), EDP \( \Pi^\alpha \) is constructed as follows, where DCA and UNA are incorporated in a first order theory \( T \) since only its Herbrand models are considered, and \( p_1, \ldots, p_\ell, z_1, \ldots, z_m, \) and \( q_1, \ldots, q_n \) are used to denote atoms from \( P, Z \) and \( Q \) respectively.

1. For any clause in \( T \) of the form:

\[ p_1 \lor \ldots \lor p_\ell \lor z_1 \lor \ldots \lor z_m \lor q_1 \lor \ldots \lor q_n \lor \neg p_{\ell+1} \lor \ldots \]
\[ \lor \neg p_s \lor \neg z_{m+1} \lor \ldots \lor \neg z_1 \lor \neg q_{n+1} \lor \ldots \lor \neg q_u, \]

\( \Pi^\alpha \) has the rule:

\[ z_1 \ldots | z_m | q_1 \ldots | q_n \leftarrow p_{\ell+1}, \ldots, p_s z_{m+1}, \ldots, z_1, q_{n+1}, \\
\ldots, q_u, \text{not} p_1, \ldots, \text{not} p_\ell. \]
2. For every clause in \( \mu Th(\Sigma) \) of the form:

\[
p_1 \lor \ldots \lor p_i \lor q_1 \lor \ldots \lor q_n \lor \neg q_{n+1} \lor \ldots \lor \neg q_u,
\]

\( \Pi^\alpha \) has the rule:

\[
p_1 | \ldots | p_i | q_1 | \ldots | q_n \leftarrow q_{n+1}, \ldots, q_u,
\]

3. For any atom \( p, z, q \), \( \Pi^\alpha \) has the rule:

\[
\neg p \leftarrow \text{notp},
\]

\[
z \mid \neg z \leftarrow, \quad q \mid \neg q \leftarrow .
\]

Remarks. It is shown in [Sakama and Inoue, 1995] that there is a 1-1 correspondence between the \( PZ \)-minimal Herbrand models of parallel circumscription and the answer sets of the translated EDP \( \Pi^\alpha \).

Definition 4.12
Let \( T \) be a first order theory without function symbols including DCA and UNA. Then given prioritized circumscription:

\[
\text{Circum}(\forall T(P^1 \ldots P^k; Z, Q); P^1 > P^2 > \ldots > P^k; Z),
\]

EDP \( \Pi^\beta \) is constructed in the following two steps:

1. This is the same as the first step given by Definition 4.3.

2. EDP \( \Pi^\beta \) consists of all rules from \((\Pi^\alpha)_1, \ldots, (\Pi^\alpha)_k \) and \( \Pi_\gamma \) where

   (a) each \((\Pi^\alpha)_i \) is an extended disjunctive program (EDP) which is constructed from the \( i \)th renamed parallel circumscription (4.1) according to Definition 4.11.

   (b) \( \Pi_\gamma \) is the same set given by Definition 4.3.

Then the relationship between prioritized circumscription and the translated EDP \( \Pi^\beta \) are given as follows.

Theorem 4.13 Let \( T \) be a first order theory without function symbols including DCA and UNA and \( F \) be a ground formula of the language of \( T \) in which equality does not occur as a predicate symbol. Then, it holds that, for any \( F \),

\[
\text{Circum}(\forall T; P^1 > P^2 > \ldots > P^k; Z) \models F
\]

iff \( F \) is true in all answer sets of EDP \( \Pi^\beta \).

where EDP \( \Pi^\beta \) is constructed from \( \text{Circum}(\forall T; P^1 > P^2 > \ldots > P^k; Z) \), by using Definition 4.12.
Example 4.14
Consider prioritized circumscription given in Example 4.10. We apply Theorem 4.13 instead of Theorem 4.6 to it. Then according to Definition 4.12, the translated EDPs \((\Pi^\alpha)_1\) and \((\Pi^\alpha)_2\) are as follows:

Since \(P = \{p1\}\), \(Z = \{q1, r1\}\), \(Q = \phi\) in the renamed circumscription: \(Circum(\{p1 \lor q1, q1 \lor r1\}; p1; q1, r1)\),

\[
(\Pi^\alpha)_1 : \quad q1 \leftarrow not p1,
q1 \leftarrow r1,
\neg p1 \leftarrow not p1,
q1 \leftarrow \neg q1, \quad r1 \leftarrow \neg r1.
\]

Since \(P = \{q2\}\), \(Z = \{r2\}\), \(Q = \{p2\}\) in the renamed circumscription: \(Circum(\{p2 \lor q2, q2 \lor r2\}; q2; r2)\),

\[
(\Pi^\alpha)_2 : \quad p2 \leftarrow not q2,
p2 \leftarrow not q2,
p2 \leftarrow q2,
neg q2 \leftarrow not q2,
r2 \leftarrow \neg r2, \quad p2 \leftarrow \neg p2.
\]

\(\Pi_\gamma\) is obtained as the same one shown in Example 4.10.
As a result, EDP \(\Pi^\beta\) has the following two answer sets:

\[
\{\neg p, q, r, \neg p1, q1, r1, \neg p2, q2, r2\},
\{\neg p, q, \neg r, \neg p1, q1, \neg r1, \neg p2, q2, \neg r2\},
\]

where \(\Pi^\beta\) consists of all rules in the above \((\Pi^\alpha)_1\), \((\Pi^\alpha)_2\) and \(\Pi_\gamma\). Both \(\neg p\) and \(q\) are true in all of these answer sets, but so is neither \(r\) nor \(\neg r\). Notice that this is the same evaluation result as the one in Example 4.10.

### 4.3 Related Works and Summary

In this chapter, we present a method of compiling circumscription into extended logic programs which is widely applicable to parallel circumscription as well as prioritized circumscription. Our method always enables us to compute any circumscription whose theory includes both DCA and UNA by compiling it into ELP \(\Pi\), which can be evaluated by using the existing logic programming interpreter such as Satoh and Iwayama’s top-down query evaluation procedure for abductive logic programming.

In the following, we compare our method with related researches from the viewpoints of the applicable class as well as the computational efficiency and feasibility. These researches are characterized by the semantics of a class of logic programs considered as the target language to which circumscription is compiled.
• Gelfond and Lifschitz’s method [1988a] as well as our previous method shown in Chapter 3 [Wakaki and Satoh, 1995], are the most efficient since they are as efficient as the evaluation of a stratified logic program. But their applicable classes are limited within a class of circumscription which has a unique model as mentioned in the introduction of this chapter.

• In Sakama and Inoue’s first method [1995], parallel circumscription is translated into a general disjunctive program (GDP). This method is applicable to a wide class of parallel circumscription, but inapplicable to prioritized circumscription though a class of GDP has the expressive power of the positive occurrences of negation as failure [Inoue and Sakama, 1994]. The most important difference between their GDP and our ELP is whether classical negation \( \neg \) is available or not. Theorem 4.13 shows that our approach exploiting classical negation can also give an extension of their alternative method whose target language is EDP, so that it may become applicable to prioritized circumscription.

As for the computational aspects, characteristic clauses should be computed in the compilation process of our method as well as their method whose time complexity is the exponential order.

• In Sakama and Inoue’s second method [1996], parallel circumscription as well as prioritized circumscription is translated into prioritized logic programs proposed by them whose declarative meaning is given by preferred answer sets defined by them. Their method, however, is immature for the purpose of the automation of circumscription since their prioritized logic program is not feasible because their method gives only the semantic aspects, but procedural issues for the query evaluation are left as their future works.

Our future work is to implement our compiling method presented in this chapter as efficiently as possible.

4.4 Proofs

Proof of Theorem 4.4 :

According to Theorem 2.11 [de Kleer and Konolige, 1989], \( \text{Circum}(T; P; Z) \) with a tuple \( Q \) of fixed predicates can be logically equivalently reduced into circumscription without fixed predicates by introducing a new tuple \( Q' \) of similar predicate symbols not in \( T \). Therefore it holds that

\[
\text{Circum}(T(P, Q, Z); P; Z) \land \bigwedge_{q \in Q, q' \in Q'} \forall x(q(x) \equiv \neg q'(x)) \\
\equiv \text{Circum}(T(P, Q, Z) \land \bigwedge_{q \in Q, q' \in Q'} \forall x(q(x) \equiv \neg q'(x)); P, Q, Q'; Z). \quad (1)
\]
According to Theorem 2.18, i.e. [Etherington, 1987a, Theorem 2], it holds that, with respect to a right-hand circumscription of (1),

\[ \text{Circum}(T(P, Q, Z) \land \bigwedge_{q \in Q, q' \in Q'} \forall x(q(x) \equiv \neg q'(x)); P, Q, Q'; Z) \models F \]

iff for any extension \( E_1 \) of a default theory \( \Delta_1 \), \( E_1 \models F \)

where

\[ \Delta_1 \overset{\text{def}}{=} \{; \neg p(x)/\neg p(x),; \neg q(x)/\neg q(x),; \neg q'(x)/\neg q'(x) \mid p \in P, q \in Q, q' \in Q' \}, \]

\[ T \land \bigwedge_{q \in Q, q' \in Q'} \forall x(q(x) \equiv \neg q'(x)) \] (2)

Now, let us define a default theory \( \Delta_2 \) as follows:

\[ \Delta_2 \overset{\text{def}}{=} \{; \neg p(x)/\neg p(x),; \neg q(x)/\neg q(x),; q(x)/q(x) \mid p \in P, q \in Q \}, \]

\[ T \land \bigwedge_{q \in Q, q' \in Q'} \forall x(q(x) \equiv \neg q'(x)) \]

Then according to the definition of the default theory, we can easily prove a 1-1 correspondence that, for any extension \( E_1 \) of a default theory \( \Delta_1 \), there exists an extension \( E_2 \) of a default theory \( \Delta_2 \) such that

\[ E_1 = E_2, \] (3)

and vice versa.

Since default rules of \( \Delta_2 \) coincide with default rules of \( \Delta \) as well as there occurs no predicate symbols of \( q' \) in both these default rules and \( T \), we can also easily prove a 1-1 correspondence that, for any extension \( E_2 \) of a default theory \( \Delta_2(P, Q, Q', Z) \), there exists an extension \( E(P, Q, Z) \) of a default theory \( \Delta \) such that

\[ \text{Th}(E_2(P, Q, Q', Z)) = \text{Th}(E(P, Q, Z) \cup \bigwedge_{q \in Q, q' \in Q'} \{ \forall x(q(x) \equiv \neg q'(x)) \}) \] (4)

and vice versa.

Therefore according to (1), (2), (3), (4), it holds that,

\[ \text{Circum}(T; P; Z) \land \bigwedge_{q \in Q, q' \in Q'} \forall x(q'(x) \equiv \neg q(x)) \models F \]

iff for any extension \( E_1 \) of a default theory \( \Delta_1 \), \( E_1 \models F \) (\( \Box \) (1), (2))

iff for any extension \( E_2 \) of a default theory \( \Delta_2 \), \( E_2 \models F \) (\( \Box \) (3))

iff for any extension \( E \) of a default theory \( \Delta \),

\[ E \land \bigwedge_{q \in Q, q' \in Q'} \forall x(q(x) \equiv \neg q'(x)) \models F \] (\( \Box \) (4))

Thus since any predicate symbol \( q' \in Q' \) does not occur in \( \text{Circum}(T; P; Z) \), \( E \) and \( F \), it holds that,

\[ \text{Circum}(T; P; Z) \models F \]

iff for any extension \( E \) of a default theory \( \Delta \), \( E \models F \) \( \Box \)
Before proving Theorem 4.5, firstly we provide Definition 4.15 and prove Lemma 4.16 as follows.

**Definition 4.15** Let $\Sigma$ be a set of ground clauses and $C$ be a characteristic clause from $\muTh(\Sigma)$. We define a set $D$ of default rules which are generated from all contrapositives of all characteristic clauses in $\muTh(\Sigma)$ as follows.

For every $C$ whose form is as follows:

$$C : \ell_1 \lor \ell_2 \lor \ldots \lor \ell_n$$

and every its contrapositive such as

$$\neg \ell_1 \land \ldots \land \neg \ell_{i-1} \land \neg \ell_{i+1} \land \ldots \land \neg \ell_n \supset \ell_i \quad (i = 1, \ldots, n),$$

$D$ has the inference rules of the following form:

$$\neg \ell_1, \ldots, \neg \ell_{i-1}, \neg \ell_{i+1}, \ldots, \neg \ell_n : \ell_i \quad (i = 1, \ldots, n)$$

where we identify $\neg (\neg p)$ with a ground atom $p$.

**Lemma 4.16** Let $\Sigma$ be a consistent set of ground clauses, $D$ be a set of default rules generated from $\Sigma$ according to Definition 4.15 and $p_i \in \text{Lit} \ (1 \leq i \leq n)$ where $\text{Lit}$ is a set of ground literals in the language of $\Sigma$. $\Delta$ and $\Delta'$ are default theories defined as follows.

$$\Delta \overset{\text{def}}{=} \left( \bigcup_{i=1}^{n} \left\{ \frac{\neg p_i}{\neg\neg p_i} \right\}, \Sigma \right)$$

$$\Delta' \overset{\text{def}}{=} \left( \bigcup_{i=1}^{n} \left\{ \frac{\neg p_i}{\neg\neg p_i} \right\} \cup D, \emptyset \right)$$

Then there is a 1-1 correspondence that for any extension $E$ of a default theory $\Delta$, there exists an extension $E'$ of a default theory $\Delta'$ such that $E \cap \text{Lit} = E' \cap \text{Lit}$, and vice versa.

**Proof:**

We prove this lemma by using a non-deterministic procedure of computing extensions [Satoh, 1992] shown in Section 2.2.2. Suppose that in Step 1 of the procedure, usable defaults are selected in the sequence of $\frac{\neg p_1}{\neg\neg p_1}, \frac{\neg p_2}{\neg\neg p_2}, \ldots$. Let $E_k$ be an extension of $\Delta$ computed at Step 2 in the $k$th iteration of the procedure as follows:

$$E_k = Th \left( E_{k-1} \cup \left\{ \frac{\neg p_k}{\neg\neg p_k} \right\} \right).$$
Then it is obvious that $E_k$ becomes also an extension of a default theory $\Delta_k$ as follows.

$$\Delta_k \overset{\text{def}}{=} \left( \bigcup_{i=1}^{k} \left\{ \vdash \neg p_i \right\}, \Sigma \right) \quad (\text{where } k \neq 0)$$

Now, corresponding to $\Delta_k$, we define $\Delta'_k$ as follows:

$$\Delta'_k \overset{\text{def}}{=} \left( \bigcup_{i=1}^{k} \left\{ \vdash \neg p_i \right\} \cup D, \phi \right)$$

and let $E'_k$ be an extension of $\Delta'_k$. Since Step 0 corresponds to $k = 0$, i.e. 0th iteration, we define $\Delta'_0$ as follows:

$$\Delta'_0 \overset{\text{def}}{=} (D, \phi)$$

and let $E'_0$ be an extension of $\Delta'_0$.

In the following, based on the induction by numbers of the iteration of the procedure, we prove $E_{k+1} \cap \text{Lit} = E'_{k+1} \cap \text{Lit}$ at the $(k+1)$th iteration by assuming $E_k \cap \text{Lit} = E'_k \cap \text{Lit}$ at the $k$th iteration, which leads to the proof of $E \cap \text{Lit} = E' \cap \text{Lit}$.

1. In case of $i = 0$, according to step 0 of the procedure, we obtain that

$E_0 = Th(\Sigma)$.

On the other hand, it holds that, $Th(\Sigma) \cap \text{Lit} = \mu Th(\Sigma) \cap \text{Lit}$.

Then, for $\forall \ell_0 \in E_0 \cap \text{Lit}$, it holds that,

$\ell_0 \in \mu Th(\Sigma)$, where $\ell_0$ is a unit literal.

Therefore, there exists a default rule in $D$ of $\Delta'_0$ whose form is $\text{true}$.\text{e}_0$.

$$\Box \ell_0 \in E'_0 \cap \text{Lit},$$

$$\Box E_0 \cap \text{Lit} \subseteq E'_0 \cap \text{Lit}.$$ 

In a similar way, we can also prove that $E'_0 \cap \text{Lit} \subseteq E_0 \cap \text{Lit}$.

Thus it is shown that, $E'_0 \cap \text{Lit} = E_0 \cap \text{Lit}$.

2. In case of $i = k$,

suppose that in the process of computing an extension $E$ of $\Delta$, usable defaults $\vdash \neg p_i \ (i = 1, \ldots, k - 1)$ have already been selected and at Step 1 of the $k$th iteration, a usable default $\vdash \neg p_k$ is selected. Then $E_k$ is obtained at the next Step 2 as follows:

$$E_k = Th(E_{k-1} \cup \left\{ \neg p_k \right\})$$

$$= Th(\Sigma \cup \left\{ \neg p_1, \ldots, \neg p_k \right\}) \quad (1)$$

Obviously $E_k$ becomes an extension of $\Delta_k$. Now according to the induction hypothesis, we assume that w.r.t. an extension $E'_k$ of $\Delta'_k$ which is constructed corresponding to $\Delta_k$, it holds that
4.4 PROOFS

\[ E_k \cap \text{Lit} = E'_k \cap \text{Lit} \]  
\[ (2) \]

Then according to (1),(2), it holds that \( \neg p_i \in E_k \) \( (i = 1, \ldots, k) \), so does 
\[ \neg p_i \in E'_k \quad (i = 1, \ldots, k). \]  
\[ (3) \]

3. In case of \( i = k + 1 \),

suppose that in the process of computing an extension \( E \) of \( \Delta \), a usable default \( \frac{\neg p_{k+1}}{\neg p_{k+1}} \) is selected at Step 1 of the \((k + 1)\)th iteration, which means \( p_{k+1} \notin E_k \). Thus according to (2), i.e. the induction hypothesis, it holds that,

\[ p_{k+1} \notin E'_k. \]  
\[ (4) \]

According to (4), since \( \frac{\neg p_{k+1}}{\neg p_{k+1}} \) becomes a usable default w.r.t.

\[ \Delta'_{k+1} = \left( \bigcup_{i=1}^{k+1} \{ \frac{\neg p_i}{\neg p_i} \} \cup D, \phi \right) \]

at Step 1 of the \((k + 1)\)th iteration of the procedure, the following result is obtained.

\[ \neg p_{k+1} \in E'_{k+1} \]  
\[ (5) \]

Thus, according to (3),(5), it holds that,

\[ \neg p_i \in E'_{k+1} \quad (i = 1, \ldots, k + 1). \]  
\[ (6) \]

Now, for any ground literal \( \ell \) such that 
\[ \ell \in E_{k+1} \cap \text{Lit} \quad \text{and} \quad \ell \notin E_k \cap \text{Lit}, \]

it holds that as follows:

\[ \Sigma \cup \{-p_1, \ldots, -p_k, -p_{k+1}\} \vdash \ell, \]  
\[ (8) \]

\[ \Sigma \cup \{-p_1, \ldots, -p_k\} \not\vdash \ell. \]

(8) denotes that \( \ell \) is inferred by using some clauses from \( \Sigma \), say \( C_i \in \Sigma \) \( (i = 1, \ldots, m) \) and \( \neg p_{j_1}, \ldots, \neg p_{j_s}, \neg p_{k+1} \) where

\[ \{-p_{j_1}, \ldots, -p_{j_s}\} \subseteq \{-p_1, \ldots, -p_k\}. \]  
\[ (9) \]

Then it holds that

\[ C_1, \ldots, C_m \vdash \neg p_{j_1} \land \ldots \land \neg p_{j_s} \land \neg p_{k+1} \supset \ell. \]

Therefore

\[ \neg p_{j_1} \land \ldots \land \neg p_{j_s} \land \neg p_{k+1} \supset \ell \in Th(\Sigma). \]

Due to \( \ell \not\in \mu Th(\Sigma) \) and \( p_{k+1} \not\in \mu Th(\Sigma) \), there exists a clause \( \tau \in \mu Th(\Sigma) \) such that includes both \( \ell \) and \( p_{k+1} \) as follows.

\[ \tau \overset{\text{def}}{=} -p'_{j_1} \land \ldots \land -p'_{j_s} \land -p_{k+1} \supset \ell \in \mu Th(\Sigma) \]
where \( \{p'_{j1}, \ldots, p'_{js}\} \subseteq \{p_{j1}, \ldots, p_{js}\} \) (10)

Therefore there exists a following default rule in \( D \) which is constructed from \( \tau \) according to Definition 4.15.

\[
\frac{\neg p'_{j1}, \ldots, \neg p'_{js}}{\ell} \in D
\] (11)

Since every rule in \( D \) is a default rule of a default theory \( \Delta'_{k+1} \) and \( \neg p_{k+1} \) as well as \( \neg p'_{jt} \) \((t = 1, \ldots, s)\) are members of \( E'_{k+1} \), (11) derives \( \ell \) as a member of the extension \( E'_{k+1} \) as follows:

\[
\ell \in E'_{k+1}.
\]

Therefore

\[
\ell \in E'_{k+1} \cap \text{Lit}
\] (12)

From (7),(12), it is proved that,

\[
E_{k+1} \cap \text{Lit} \subseteq E'_{k+1} \cap \text{Lit}.
\]

In a similar way, we can prove as follows.

\[
E'_{k+1} \cap \text{Lit} \subseteq E_{k+1} \cap \text{Lit},
\]

Finally we can obtain the result as follows:

\[
E_{k+1} \cap \text{Lit} = E'_{k+1} \cap \text{Lit}
\]

\[
\square
\]

**Proof of Theorem 4.5:**

In case that \( T \) (i.e. \( \Sigma \)) is inconsistent, it is easily shown that \( \text{Circum}(T; P; Z) \models G \) for any \( G \in \text{Lit} \). On the other hand, there exists an integrity constraint \( \leftarrow \in \Pi_\alpha \) because of \( \mu \text{Th}(\Sigma) = \{ \Box \} \). Therefore the answer set of \( \Pi_\alpha \) has a unique answer set, \( \text{Lit} \), which means that, for any \( G \in \text{Lit} \), \( G \) is true in the unique answer set \( \text{Lit} \) of \( \Pi_\alpha \). Thus Theorem 4.5 is proved.

Next, we consider the case that \( T \) (i.e. \( \Sigma \)) is consistent as follows. According to Theorem 4.4, for any ground literal \( G \) of the language of \( T \) whose predicate symbol is not equality, it holds that,

\[
\text{Circum}(T; P; Z) \models G
\]

iff for any extension \( E \) of a default theory \( \Delta \), \( E \models G \) iff for any extension \( E \) of a default theory \( \Delta \), \( E \models G \) (1)

where \( \Delta \) is defined as follows:

\[
\Delta \overset{\text{def}}{=} \left\{ \begin{array}{l}
\{ : \neg p(x) : \neg q(x) : q(x) \mid p \in P, q \in Q \} \cup T
\end{array} \right\}
\]

By the way, according to (4.1),

\[
T \overset{\text{def}}{=} \Sigma \cup \text{DCA} \cup \text{UNA} \equiv \Sigma^{\text{DCA}} \cup \text{DCA} \cup \text{UNA}
\]

where \( \Sigma^{\text{DCA}} \) is a set of ground clauses obtained by replacing all variables in each clause in \( \Sigma \) by every ground term of \( \text{DCA} \).
Now, we define a default theory $\Delta_1$ as follows:

$$\Delta_1 \overset{\text{def}}{=} \left\{ \frac{\neg p(x)}{p \in P}, \frac{\neg q(x)}{q \in Q} \right\} \cup \Sigma_{DCA}$$

Then we can easily prove that for any extension $E$ of $\Delta$, there exists an extension $E_1$ of $\Delta_1$ which corresponds to $E$ with a 1-1 correspondence such that

$$Th(E) = Th(E_1 \cup DCA \cup UNA)$$

(2)

and vice versa.

Next, let $\Delta_2$ be a default theory defined as follows:

$$\Delta_2 \overset{\text{def}}{=} \left\{ \frac{\neg p(x)}{p \in P}, \frac{\neg q(x)}{q \in Q} \right\} \cup \Sigma_{DCA}$$

where $D$ is a set of default rules each of which is generated from each characteristic clause in $\mu Th(\Sigma_{DCA})$ according to Definition 4.15.

Then according to Lemma 4.16, for any extension $E_1$ of $\Delta_1$, there exists an extension $E_2$ of $\Delta_2$ which corresponds to $E_1$ with a 1-1 correspondence such that

$$E_1 \cap \text{Lit} = E_2 \cap \text{Lit}$$

(3)

and vice versa.

According to Theorem 2.26 [Gelfond and Lifschitz, 1991, Proposition 3], for any extension $E_2$ of a default theory $\Delta_2$, there exists an answer set $S$ of ELP $\Pi^\alpha$ with a 1-1 correspondence such that

$$E_2 \cap \text{Lit} = S$$

(4)

and vice versa.

Therefore, based on the results of (1)~(4), for any ground literal $G$ of the language of $T$ whose predicate symbol is not equality, it holds that

$$\text{Circum}(\forall T; P; Z) \models G$$

iff for any extension $E$ of a default theory $\Delta$, $G \in E$ (according to (1))

iff for any extension $E_1$ of a default theory $\Delta_1$, $G \in E_1 \cup DCA \cup UNA$ (according to (2))

iff for any extension $E_1$ of a default theory $\Delta_1$, $G \in E_1$ (because the predicate symbol of $G$ is not equality.)

iff for any extension $E_2$ of a default theory $\Delta_2$, $G \in E_2$ (according to (3))

iff for any answer set $S$ of ELP $\Pi^\alpha$, $G \in S$ (according to (4))

$\Box$

In the following, we firstly prepare some definitions and lemmas, and then prove Theorem 4.13.

**Definition 4.17** Let $\Pi$ be an extended disjunctive program. We write $\text{AS}(\Pi)$ as a set of all answer sets of $\Pi$. We say an answer set of $\Pi$ is **consistent** if every atom $p$ and its negation $\neg p$, both is not simultaneously in the answer set. For any ground formula $\phi$, we write $\Pi \models \phi$ iff $\phi$ is true in every answer set of $\Pi$. 
Theorem 4.18 [Sakama and Inoue, 1995] Let $\Pi^\alpha$ be an extended disjunctive program constructed from Circum($\forall T; P; Z$) by using definition 4.11 and $F$ be a ground formula. Then it holds that

$$\text{Circum}(\forall T; P; Z) \models F \iff \Pi^\alpha \models F$$

Remarks. It is proved that there is a 1-1 correspondence between the Herbrand models of Circum($\forall T; P; Z$) and the answer sets of the translated EDP $\Pi^\alpha$ where DCA and UNA are incorporated in $T$. That is, $S$ is a $PZ$-minimal model of $\forall T$ iff $S$ is an answer set of the translated EDP $\Pi^\alpha$. The positive and negative literals contained explicitly in an answer set $S$ of EDP $\Pi^\alpha$ represent elements which are true and false respectively in the corresponding $PZ$-minimal model $S$ (see [Sakama and Inoue, 1995]).

Lemma 4.19 Suppose that $\forall T$ is consistent. Let $(\Pi^\alpha)_i$ be an EDP defined by Definition 4.11, and $(\Pi_\gamma)_i$, $(\Pi^\alpha_\gamma)_i$, and $(\Pi^\beta)_i$ be sets defined as follows.

$$(\Pi_\gamma)_i \overset{def}{=} \{ L \leftarrow L_i \mid L \leftarrow L_i \in \Pi_\gamma \text{ and } L_i \text{ is the } i\text{-th name changed literal.} \}.$$  

$$(\Pi^\alpha_\gamma)_i \overset{def}{=} ((\Pi^\alpha)_i \cup (\Pi_\gamma)_i)$$  

$$(\Pi^\beta)_i \overset{def}{=} \bigcup_{j=1}^i ((\Pi^\alpha_\gamma)_j)$$

Then it holds that

$$\text{AS}((\Pi^\beta)_i) = \{ S^\alpha_\gamma \cup S^\beta_{i-1} \mid S^\alpha_\gamma \cup S^\beta_{i-1} \text{ is consistent where } S^\alpha_\gamma \in \text{AS}((\Pi^\alpha_\gamma)_i) \text{ and } S^\beta_{i-1} \in \text{AS}((\Pi^\beta)_{i-1}) \}.$$
\[ \sigma_i^{\alpha j} \land \sigma_{i-1}^{\beta k} \models \phi. \]  
(2)

In (2), if \( \sigma_i^{\alpha j} \land \sigma_{i-1}^{\beta k} \) is consistent, \( \sigma_i^{\alpha j} \) and \( \sigma_{i-1}^{\beta k} \) should be equivalent each other since each of them represents the Herbrand model of \( T \).

Now let us define \( \phi_1 \) and \( \phi_2 \) as follows.

\[
\begin{align*}
\phi_1 & \overset{\text{def}}{=} \sigma_i^{\alpha j} \lor \ldots \lor \sigma_i^{\alpha n}, \\
\phi_2 & \overset{\text{def}}{=} \sigma_{i-1}^{\beta j} \lor \ldots \lor \sigma_{i-1}^{\beta m}.
\end{align*}
\]

Then it holds that \( (\Pi^{\alpha j})_{i} \models \phi_1 \) and \( (\Pi^{\beta j})_{i-1} \models \phi_2 \) and \( \phi_1 \land \phi_2 \models \phi. \)

\[ \iff \] By Lemma 4.19, \( (\Pi^{\beta j})_{i} \models \phi_1 \) and \( (\Pi^{\beta j})_{i} \models \phi_2 \). Therefore, \( (\Pi^{\beta j})_{i} \models \phi. \)

**Definition 4.21** We define Circum\(_i\) and ACircum\(_i\) as follows:

\[
\begin{align*}
\text{Circum}_i & \overset{\text{def}}{=} \text{Circum}(\forall T; P^n, P^{n+1}, \ldots, P^k; Z), \\
\text{ACircum}_i & \overset{\text{def}}{=} \bigwedge_{j=1}^i \text{Circum}_j.
\end{align*}
\]

Then according to Theorem 2.10, it holds that

\[ \text{Circum}(\forall T; P^1 > P^2 > \ldots > P^k; Z) \equiv \text{ACircum}_k \]

**Theorem 4.13** Let \( F \) be a ground formula of the language of \( T \) and equality does not occur as a predicate symbol in \( F \). Then, it holds that, for any \( F \) and every \( i = 1, \ldots, k \),

\[ \text{ACircum}_i \models F \iff (\Pi^{\beta j})_{i} \models F \]

**Proof:**

In case that \( T \) is inconsistent, it is easily proved that this theorem holds.

So, when \( T \) is consistent, we prove the theorem by the induction of level of \( \text{Circum}_i \) as follows.

1. In case of \( i=1 \), according to Theorem 4.18,

\[ \text{ACircum}_1 \models F \iff (\Pi^{\beta j})_{1} \models F \]

2. By assuming that this Theorem holds in case of \( i=j-1 \), we show that this Theorem also holds in case of \( i=j \). According to Lemma 4.20, it holds that \( (\Pi^{\beta j})_{j} \models F \) iff there exist \( F_1, F_2 \) such that \( (\Pi^{\alpha j})_{j} \models F_1 \) and \( (\Pi^{\beta j})_{j-1} \models F_2 \) and \( F_1 \land F_2 \models F \).

\[ \therefore \] According to Theorem 4.18, it holds that

\[ (\Pi^{\alpha j})_{j} \models F_1 \iff \text{Circum}_j \models F_1. \]

By induction hypothesis,

\[ (\Pi^{\beta j})_{j-1} \models F_2 \iff \text{ACircum}_{j-1} \models F_2. \]

Then according to \( \text{ACircum}_j = \text{ACircum}_{j-1} \land \text{Circum}_j \) and (1), (2), (3), this theorem is proved. \( \square \)