ICOT Technical Report: TR-0825

TR-0825

Negation in Disjunctive Logic Programs

by

C. Sakawa (ASTEM) & K. Inoue

December, 1992

Negation in Disjunctive Logic Programs

Chiaki Sakama
ASTEM
Research Institute of Kyoto
17 Chudoji Minami-machi
Shimogyo, Kyoto 600, Japan
sakama@astem.or.jp

Katsumi Inoue
ICOT
Mita-Kokusai Bldg., 21F
1-4-28 Mita, Minato-ku
Tokyo 108, Japan
inoue@icot.or.jp

November 29, 1992

Abstract

In this paper, we study inferring negation from disjunctive logic programs. First, we consider extensions of the GCWA and the WGCWA for general disjunctive programs based upon the stable model semantics. We define new rules the GCWA$^{\neg}$ and the WGCWA$^{\neg}$ which are natural extensions of the GCWA and the WGCWA. Second, we introduce a new semantics called the *possible world semantics* for general disjunctive programs, which was initially introduced in [Sak89] for positive disjunctive programs. The *possible world assumption (PWA)* infers negation under the possible world semantics, which lies between the GCWA and the WGCWA in positive disjunctive programs. The PWA is also extended to the PWA$^{\neg}$ for general disjunctive programs. Then it is shown that the PWA$^{\neg}$ provides the most careful negative inference compared with the GCWA$^{\neg}$ and the WGCWA$^{\neg}$. We also present a bottom-up model generation proof procedure to compute each negation in general disjunctive programs.

1 Introduction

In logic programming and deductive databases, Reiter's *closed world assumption (CWA)* [Rei78] is usually employed as a default rule for inferring negation from a program. However, it is also well-known that the CWA works well only for definite Horn programs and causes an inconsistency in the presence of disjunctive information in a program. In the context of disjunctive logic programming, Reiter's CWA is mainly extended in two ways: one is Minker's GCWA [Min82] and the other is Rajasekar et al's WGCWA [RLM89] (or equivalently the DDR [RT88]). The GCWA is based upon the *minimal model semantics* of disjunctive programs and usually interprets disjunctions exclusively, while the WGCWA is weaker than the GCWA and interprets disjunctions inclusively. The problem is that both the GCWA and the WGCWA fairly extend the CWA, but they are inherently *different* rules in their own rights. In fact, in the absence of a single uniform framework, one has to use different rules to treat both exclusive and inclusive disjunctions in the same program. Such a situation actually happens; for instance, consider the situation that "Calendar days are usually classified into Sundays, National-holidays and other weekdays. In Japan, when a National-holiday falls on Sunday, the holiday is transferred to Monday." This situation is presented in the program:

$$Sunday \vee National\text{-}holiday \vee Weekday$$

$$Monday\text{-}is\text{-}holiday \leftarrow Sunday \wedge National\text{-}holiday$$

in which $Sunday \vee Weekday$ is exclusive, while $Sunday \vee National\text{-}holiday$ is inclusive.

To treat such a situation, Chan and Sakama [Cha89, Sak89] have proposed the *possible world semantics (PWS)*. In a positive disjunctive program, the *possible world assumption (PWA)* infers negation under the possible world semantics. The PWA lies between the GCWA and the WGCWA, and can distinguish both types of disjunctions in a uniform manner.

In this paper, we firstly extend the GCWA and the WGCWA to the GCWA$^{\neg}$ and the WGCWA$^{\neg}$ for general disjunctive programs. These are natural extensions of the corresponding rules and defined through the stable model semantics of general disjunctive programs. Next, we extend the PWA to the PWA$^{\neg}$ in general disjunctive programs. Compared with the GCWA$^{\neg}$ and the WGCWA$^{\neg}$, the PWA$^{\neg}$ enjoys several nice features. Finally, we also present an algorithm to compute each negation in general disjunctive programs.

The rest of this paper is organized as follows. In Section 2, we review the previously proposed results on positive disjunctive programs. In Section 3, we extend these results to general disjunctive programs. In Section 4, we present a method for computing each negation using a bottom-up model generation proof procedure.

2 Negation in Positive Disjunctive Programs

2.1 Positive Disjunctive Programs

A *positive disjunctive program* is a finite set of clauses of the form:

$$A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m \;\; (l, m \geq 0)$$

where A_i's and B_j's are atoms and all variables are assumed to be universally quantified at the front of the clause. A clause is called *disjunctive* (resp. *definite*, *negative*) if $l > 1$ (resp. $l = 1$, $l = 0$). A program containing only definite clauses is called a *definite program* and a program containing definite and possibly negative clauses is called a *Horn program*. The disjunction $A_1 \vee \ldots \vee A_l$ is called the *head* and the conjunction $B_1 \wedge \ldots \wedge B_m$ is called the *body* of the clause. A *ground clause* is a clause which contains no variable. A *ground program* is a program in which every variable is instantiated by the elements of the Herbrand universe of the program in every possible way. A ground program is a possibly infinite set of ground clauses. From the semantical point of view, a program is equivalent to its ground program, thus we consider a ground program in this paper unless stated otherwise.

An *interpretation* of a program P is a subset of the Herbrand base $\mathcal{HB}_P$ of the program. An interpretation I *satisfies* the clause $A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m$ if $B_1, \ldots, B_m \in I$ implies $A_i \in I$ for some i $(1 \leq i \leq l)$. Especially, if there is a clause such that $l = 0$ and $B_1, \ldots, B_m \in I$, I does *not* satisfy the negative clause. For a program P, a minimal set I which satisfies every clause in P is called a *minimal model* of P. If P has a unique minimal model I, it is also called the *least Herbrand model*. When there exists a minimal model of P, P is called *consistent*; otherwise, it is called *inconsistent*.

2.2 GCWA, WGCWA and PWA

For inferring negation from positive disjunctive programs, two alternative extensions of the CWA are well known. One is the *generalized closed world assumption (GCWA)* proposed by Minker [Min82], and the other is the *weak generalized closed world assumption (WGCWA)* by Rajasekar et al [RLM89].

Definition 2.1 [Min82] Let P be a consistent positive disjunctive program and $\mathcal{MM}_P$ be the set of all minimal models of P. Then $GCWA(P)$ is defined by the set:

$$GCWA(P) = \{\neg A \mid A \in \mathcal{HB}_P \text{ and } A \notin I \text{ for any } I \in \mathcal{MM}_P\}. \quad \square$$

The *Horn translation* [RT88] of a disjunctive program P is defined by:

$$Horn(P) = \{A_i \leftarrow B_1 \wedge \ldots \wedge B_m \mid A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m \in P \text{ and } 1 \leq i \leq l\}.$$

Note here that $Horn(P)$ is always consistent, since it does not contain negative clauses.

Definition 2.2 [RT88, RLM89][1] Let P be a consistent positive disjunctive program and $Horn(P)$ be its Horn translation. Suppose that $M_{Horn(P)}$ is the least Herbrand model of $Horn(P)$. Then $WGCWA(P)$ is defined by the set:

$$WGCWA(P) = \{\neg A \mid A \in \mathcal{HB}_P \text{ and } A \notin M_{Horn(P)}\}. \quad \Box$$

Properties of the GCWA and the WGCWA are as follows.

Theorem 2.1 [Min82, RLM89] Let P be a consistent positive disjunctive program and A be a ground atom. Then,

(i) $P \cup GCWA(P)$ is consistent.
$P \cup WGCWA(P)$ is consistent.

(ii) $P \models A$ iff $P \cup GCWA(P) \models A$.
$P \models A$ iff $P \cup WGCWA(P) \models A$.

(iii) $P \subseteq P'$ does not imply $GCWA(P') \subseteq GCWA(P)$.
$P \subseteq P'$ implies $WGCWA(P') \subseteq WGCWA(P)$.

(iv) $WGCWA(P) \subseteq GCWA(P)$.

(v) For a definite program P, $GCWA(P) = WGCWA(P) = CWA(P)$. $\Box$

That is, (i) both $GCWA(P)$ and $WGCWA(P)$ are *consistent* with P, (ii) positive facts proven from P are *invariant*, (iii) the GCWA (resp. WGCWA) is *nonmonotonic* (resp. *monotonic*), (iv) the GCWA is *stronger* than the WGCWA, and (v) for definite programs each rule *reduces* to the CWA.

Example 2.1 Let $P = \{a \vee b \leftarrow,\ c \leftarrow a \wedge b\}$. Then $GCWA(P) \models \neg c$ and $WGCWA(P) \not\models \neg c$. $\Box$

In the above example, the difference between $GCWA(P)$ and $WGCWA(P)$ comes from the interpretation of $a \vee b$. That is, $GCWA(P)$ interprets the disjunction exclusively, while $WGCWA(P)$ interprets it inclusively. However, consider the program $P' = P \cup \{\leftarrow a \wedge b\}$. In this new program P', the clause $\leftarrow a \wedge b$ inhibits an inclusive interpretation of $a \vee b$, then $\neg c$ should be true, while $WGCWA(P')$ still cannot infer $\neg c$. This is because the WGCWA does not consider the effect of negative clauses in a program and often fails to capture the

[1] Here we employ the definition by Ross and Topor [RT88] who have introduced it in the context of the *disjunctive database rule (DDR)*. According to [RLM89, LMR92], the DDR and the WGCWA are equivalent.

intended meaning of the program. Generally speaking, the GCWA is too strong to interpret inclusive disjunctions, while the WGCWA is too weak to treat exclusive disjunctions. Then to treat both types of disjunctions in a program, one has to use different rules in the same program.

To improve such a situation, Chan and Sakama [Cha89, Sak89] have proposed the *possible world semantics (PWS)* which can distinguish both types of disjunctions in a uniform manner. The following results are due to [Sak89].

Given a ground disjunctive clause $C : A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m$ and a non-empty subset S of $\{A_1, .., A_l\}$, the *split* of C with respect to S is defined by the set of ground Horn clauses $\{A_i \leftarrow B_1 \wedge \ldots \wedge B_m \mid A_i \in S\}$. Here, C has $2^l - 1$ splits.

Definition 2.3 Let P be a positive disjunctive program. Then $\mathbf{Horn}(P)$ is the set of all ground Horn programs such that each Horn program P' in $\mathbf{Horn}(P)$ is obtained by

(i) replacing each ground disjunctive clause from P with the clauses in one of its splits;

(ii) keeping other (non-disjunctive) ground clauses from P. □

Definition 2.4 Let P be a positive disjunctive program. Then the set of *possible worlds* $\mathcal{PW}_P$ of P is defined by the set of least Herbrand models of consistent programs in $\mathbf{Horn}(P)$. □

Example 2.2 Let $P = \{a \vee b \leftarrow,\ b \vee c \leftarrow,\ \leftarrow b \wedge c\}$. Then $\mathbf{Horn}(P) = \{\{a \leftarrow,\ b \leftarrow,\ \leftarrow b \wedge c\}, \{a \leftarrow,\ c \leftarrow,\ \leftarrow b \wedge c\}, \{b \leftarrow,\ \leftarrow b \wedge c\}, \{b \leftarrow,\ c \leftarrow,\ \leftarrow b \wedge c\}, \{a \leftarrow,\ b \leftarrow,\ c \leftarrow,\ \leftarrow b \wedge c\}\}$. Since the last two of $\mathbf{Horn}(P)$ are inconsistent, the set of possible worlds of P is $\mathcal{PW}_P = \{\{a, b\}, \{a, c\}, \{b\}\}$. □

Lemma 2.2 [Sak89] A consistent positive disjunctive program has at least one possible world. □

Lemma 2.3 [Sak89] A possible world of a positive disjunctive program P is a model of P. □

The notion of possible worlds is different from minimal models. In fact, in Example 2.2 $\{a, b\}$ is a possible world, but not a minimal model. By definition, the set of all possible worlds includes the set of all minimal models.

Lemma 2.4 [Sak89] Let P be a consistent positive disjunctive program, $\mathcal{MM}_P$ be the set of all minimal models of P, and $\mathcal{PW}_P$ be the set of all possible worlds of P. Then the set of all minimal elements from $\mathcal{PW}_P$ coincides with $\mathcal{MM}_P$. □

Especially, a definite program has a unique possible world which is the least Herbrand model of the program. Under the possible world semantics, negation is defined as follows.

Definition 2.5 Let P be a consistent positive disjunctive program and $\mathcal{PW}_P$ be the set of all possible worlds of P. Then the *possible world assumption (PWA)* of P is defined by the set:

$$PWA(P) = \{\neg A \mid A \in \mathcal{HB}_P \text{ and } A \notin I \text{ for any } I \in \mathcal{PW}_P\}. \quad \Box$$

Theorem 2.5 [Sak89] Let P be a consistent positive disjunctive program and A be a ground atom. Then,

(i) $P \cup PWA(P)$ is consistent.
(ii) $P \models A$ iff $P \cup PWA(P) \models A$.
(iii) $P \subseteq P'$ does not imply $PWA(P') \subseteq PWA(P)$.
(iv) For a definite program P, $PWA(P) = CWA(P)$. $\Box$

The next theorem presents that the PWA is stronger than the WGCWA and weaker than the GCWA.

Theorem 2.6 [Sak89] Let P be a consistent positive disjunctive program. Then $WGCWA(P) \subseteq PWA(P) \subseteq GCWA(P)$ holds. Especially, if $P \cup Horn(P)$ is consistent, $WGCWA(P) = PWA(P)$. $\Box$

Example 2.3 (cont. from Example 2.1) Let $P = \{a \vee b \leftarrow,\ c \leftarrow a \wedge b\}$ and $P' = P \cup \{\leftarrow a \wedge b\}$. Then, $PWA(P) \not\models \neg c$, while $PWA(P') \models \neg c$. $\Box$

It should be noted that $P' \cup Horn(P')$ is inconsistent in the above example, hence $PWA(P')$ infers proper negation compared with $WGCWA(P')$.

3 Negation in General Disjunctive Programs

3.1 General Disjunctive Programs

A *general disjunctive program* is a finite set of clauses of the form:

$$A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m \wedge not B_{m+1} \wedge \ldots \wedge not B_n \quad (l \geq 0,\ n \geq m \geq 0)$$

where A_i's and B_j's are atoms and all variables are assumed to be universally quantified at the front of the clause. An operator *not* preceded each atom $B_k (m+1 \leq k \leq n)$ denotes *negation by failure* [Cla78]. A clause is called *disjunctive* (resp. *normal*, *negative*), if $l > 1$ (resp. $l = 1$, $l = 0$). A program containing only normal clauses is called a *normal program* and a program containing normal and possibly negative clauses is called a *general logic program*. A program which contains no predicate defined recursively through its negation is called *stratified*. A general disjunctive program reduces to a positive disjunctive program, when $m = n$ (containing no *not*) for every clause. The notion of head, body, ground clause (program) are defined in the same way as in the previous section.

An interpretation I *satisfies* the clause $A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m \wedge not B_{m+1} \wedge \ldots \wedge not B_n$ if $B_1, \ldots, B_m \in I$ and $B_{m+1}, \ldots, B_n \notin I$ implies $A_i \in I$ for some i $(1 \leq i \leq l)$. Especially, if there is a clause such that $l = 0$, $B_1, \ldots, B_m \in I$ and $B_{m+1}, \ldots, B_n \notin I$, I does *not* satisfy the negative clause. An interpretation which satisfies every clause in a program is called a *model* of the program.

As for the semantics of general disjunctive programs, we consider the *stable model semantics* of disjunctive programs which was initially introduced by Gelfond and Lifschitz [GL88] for normal programs.

Definition 3.1 Let P be a general disjunctive program and I be its interpretation. Consider a positive disjunctive program P^I obtained from P as follows:

$$\begin{aligned} P^I \;=\; & \{A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m \mid \text{ there is a ground clause } A_1 \vee \ldots \vee A_l \leftarrow \\ & B_1 \wedge \ldots \wedge B_m \wedge not B_{m+1} \wedge \ldots \wedge not B_n \; (l \geq 0) \text{ from } P \text{ and } B_{m+1}, \ldots, B_n \notin I\}. \end{aligned}$$

Then if I coincides with a minimal model of P^I, I is called a *stable model* of P. □

Note that the above definition is an extension of the original one in the sense that stable models are defined for programs containing disjunctive clauses as well as negative clauses. Similar extension is also found in [Prz90a]. We say that a general disjunctive program is *consistent* if it has a stable model; otherwise, it is called *inconsistent*.

3.2 GCWA$^\neg$, WGCWA$^\neg$ and PWA$^\neg$

In this section, we extend the GCWA, the WGCWA, and the PWA to general disjunctive programs.

Definition 3.2 Let P be a consistent general disjunctive program and $\mathcal{ST}_P$ be the set of all stable models of P. Then $GCWA^\neg(P)$ is defined by the set:

$$GCWA^\neg(P) = \{\neg A \mid A \in \mathcal{HB}_P \text{ and } A \notin I \text{ for any } I \in \mathcal{ST}_P\}. \quad □$$

To define a suitable extension of the WGCWA, we introduce a translation which transforms a general disjunctive program into a normal program.

Definition 3.3 The *normal translation* of a general disjunctive program P is defined by:

$$NP(P) = \{A_i \leftarrow B_1 \wedge \ldots \wedge B_m \wedge not B_{m+1} \wedge \ldots \wedge not B_n \mid A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m \wedge not B_{m+1} \wedge \ldots \wedge not B_n \in P \text{ and } 1 \leq i \leq l\}. \quad \square$$

The $NP(P)$ is a direct extension of $Horn(P)$, but it is not always consistent.

Example 3.1 Let $P = \{a \vee b \leftarrow not\, a\}$. Then $\mathcal{ST}_P = \{\{b\}\}$, while $\mathcal{ST}_{NP(P)} = \emptyset$. $\square$

Definition 3.4 Let P be a consistent general disjunctive program and $NP(P)$ be its normal translation. Let $\mathcal{ST}_P$ and $\mathcal{ST}_{NP(P)}$ be the sets of all stable models of P and $NP(P)$, respectively. Then $WGCWA^{\neg}(P)$ is defined by the set:

$$WGCWA^{\neg}(P) = \{\neg A \mid A \in \mathcal{HB}_P \text{ and } A \notin I \text{ for any } I \in \mathcal{ST}_P \cup \mathcal{ST}_{NP(P)}\}. \quad \square$$

Next we define the possible world semantics for general disjunctive programs.

Given a ground disjunctive clause $C: A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots B_m \wedge not B_{m+1} \wedge \ldots \wedge not B_n$ and a non-empty subset S of $\{A_1, .., A_l\}$, the *split* of C with respect to S is defined by the set of ground clauses $\{A_i \leftarrow B_1 \wedge \ldots \wedge B_m \wedge not B_{m+1} \wedge \ldots \wedge not B_n \mid A_i \in S\}$. Here, C has $2^l - 1$ splits.

Definition 3.5 Let P be a general disjunctive program. Then $\mathbf{GLP}(P)$ is the set of all ground general logic programs such that each general logic program P' in $\mathbf{GLP}(P)$ is obtained by

(i) replacing each ground disjunctive clause from P with the clauses in one of its splits;

(ii) keeping other (non-disjunctive) ground clauses from P. $\square$

Definition 3.6 Let P be a general disjunctive program. The set of *possible worlds* $\mathcal{PW}_P$ of P is defined by the set of stable models of consistent general logic programs in $\mathbf{GLP}(P)$. $\square$

Lemma 3.1 A consistent general disjunctive program has at least one possible world. $\square$

Lemma 3.2 A possible world of a general disjunctive program P is a model of P. $\square$

Lemma 3.3 Let P be a consistent general disjunctive program, $\mathcal{ST}_P$ be the set of all stable models of P, and $\mathcal{PW}_P$ be the set of all possible worlds of P. Then the set of all minimal elements from $\mathcal{PW}_P$ coincides with $\mathcal{ST}_P$.

Proof: By definition, a stable model M of P is also a stable model of some split program in **GLP**(P). Then M is also a possible world of P. Since M is minimal, it is also a minimal element in $\mathcal{PW}_P$. On the other hand, if M is minimal in $\mathcal{PW}_P$, it is also a minimal model of P (by Lemma 3.2). By the definition of possible worlds, it is also stable. □

Especially, possible worlds coincide with stable models in general logic programs.

Definition 3.7 Let P be a general disjunctive program and $\mathcal{PW}_P$ be the set of all possible worlds of P. Then $PWA^{\neg}(P)$ is defined by the set:

$$PWA^{\neg}(P) = \{\neg A \mid A \in \mathcal{HB}_P \text{ and } A \notin I \text{ for any } I \in \mathcal{PW}_P\}. \quad \square$$

Now we investigate properties of each rule. In the following, $P \models_{ST} A$ (resp. $P \models_{PW} A$) iff for any $I \in \mathcal{ST}_P$ (resp. $I \in \mathcal{PW}_P$), $I \models A$.

Theorem 3.4 Let P be a consistent general disjunctive program and A be a ground atom. Then the following properties hold.

1. (i) $P \cup GCWA^{\neg}(P)$ is consistent.
   (ii) $P \models_{ST} A$ iff $P \cup GCWA^{\neg}(P) \models_{ST} A$.
   (iii) $P \subseteq P'$ does not imply $GCWA^{\neg}(P') \subseteq GCWA^{\neg}(P)$.
   (iv) For a positive disjunctive program P, $GCWA^{\neg}(P) = GCWA(P)$.
2. (i) $P \cup WGCWA^{\neg}(P)$ is consistent.
   (ii) $P \models_{ST} A$ iff $P \cup WGCWA^{\neg}(P) \models_{ST} A$.
   (iii) $P \subseteq P'$ does not imply $WGCWA^{\neg}(P') \subseteq WGCWA^{\neg}(P)$.
   (iv) For a positive disjunctive program P, $WGCWA^{\neg}(P) = WGCWA(P)$.
3. (i) $P \cup PWA^{\neg}(P)$ is consistent.
   (ii) $P \models_{PW} A$ iff $P \cup PWA^{\neg}(P) \models_{PW} A$.
   (iii) $P \subseteq P'$ does not imply $PWA^{\neg}(P') \subseteq PWA^{\neg}(P)$.
   (iv) For a positive disjunctive program P, $PWA^{\neg}(P) = PWA(P)$.

Proof: 1. (i) Since P is consistent, it has at least one stable model and every negated atom in $GCWA^{\neg}(P)$ is not in any stable model of P, hence $P \cup GCWA^{\neg}(P)$ is consistent. (ii) For invariance of positive facts, if $P \cup GCWA^{\neg}(P) \models_{ST} A$, A is true in every stable model, hence $P \models_{ST} A$. The converse is also true. (iii) Nonmonotonicity is clear since the GCWA$^{\neg}$ includes the GCWA (by (iv)) which is nonmonotonic. (iv) Since stable models coincide with minimal models in a positive disjunctive program, the result immediately follows.

2. (i) Consistency of $WGCWA^{\neg}(P)$ follows from the fact that $\mathcal{ST}_P \subsetneq \mathcal{ST}_P \cup \mathcal{ST}_{NP(P)}$ and any atom assumed false under $WGCWA^{\neg}(P)$ is not included in any stable model of P. (ii) The result also follows from the proof of (i). (iii) For nonmonotonicity, see Example 3.2. (iv) Since $\mathcal{ST}_P \cup \mathcal{ST}_{NP(P)}$ reduces to $\mathcal{MM}_P \cup M_{Horn(P)}$ in a positive disjunctive program P, and each minimal model in $\mathcal{MM}_P$ is a subset of $M_{Horn(P)}$, the result also holds. The part 3 is proved in a similar way to part 1. □

Note that in contrast to the WGCWA, the WGCWA$^{\neg}$ is nonmonotonic.

Example 3.2 Let $P_1 = \{a \vee b \leftarrow not\, c,\;\; c \leftarrow d\}$ and $P_2 = P_1 \cup \{d \leftarrow\}$. Then $WGCWA^{\neg}(P_1) \models \neg c$ and $\neg d$, while $WGCWA^{\neg}(P_2) \models \neg a$ and $\neg b$. □

For consistent general logic programs, the three rules coincide with each other.

Lemma 3.5 Let P be a consistent general logic program. Then, $GCWA^{\neg}(P) = WGCWA^{\neg}(P) = PWA^{\neg}(P)$.

Proof: For a consistent general logic program P, $\mathcal{ST}_P \cup \mathcal{ST}_{NP(P)} = \mathcal{ST}_P$. Then the relation $GCWA^{\neg}(P) = WGCWA^{\neg}(P)$ holds by each definition. The relation $WGCWA^{\neg}(P) = PWA^{\neg}(P)$ also holds since $\mathcal{ST}_P = \mathcal{PW}_P$ for a consistent general logic program P. □

The next theorem presents the relationship between each rule in general disjunctive programs.

Theorem 3.6 Let P be a consistent general disjunctive program. Then,

(i) $WGCWA^{\neg}(P) \subseteq GCWA^{\neg}(P)$.
(ii) $PWA^{\neg}(P) \subseteq GCWA^{\neg}(P)$.

Proof: When P is consistent, $\mathcal{ST}_P \subseteq \mathcal{ST}_P \cup \mathcal{ST}_{NP(P)}$ by definition, hence (i) follows. The part (ii) also follows from the fact that $\mathcal{ST}_P \subseteq \mathcal{PW}_P$. □

As for the WGCWA$^{\neg}$ and the PWA$^{\neg}$, there is no inclusion relationship.

Example 3.3 Let $P = \{a \vee b \vee c \leftarrow not\, d,\ \ e \leftarrow a \wedge b \wedge not\, c\}$. Then $\mathcal{ST}_P = \{\{a\},\{b\},\{c\}\}$ and $\mathcal{ST}_{NP(P)} = \{\{a,b,c\}\}$, hence $WGCWA^{\neg}(P) \models \neg d$ and $\neg e$. While, there is a possible world $\{\{a,b,e\}\}$, then $PWA^{\neg}(P) \not\models \neg e$. Hence $WGCWA^{\neg}(P) \not\subseteq PWA^{\neg}(P)$. Clearly, the converse inclusion relation does not hold by Theorem 2.6 either, since each rule reduces to the PWA and the WGCWA in positive disjunctive programs. □

In the above example, $WGCWA^{\neg}(P)$ treats the disjunction $a \vee b \vee c$ inclusively, then it infers $\neg e$. This is also the case for $GCWA^{\neg}(P)$ which treats it exclusively. On the other hand, there is a possible world in which a and b are inclusively true and c is exclusively false at the same time, then $\neg e$ is not inferred by $PWA^{\neg}(P)$. This example illustrates that the possible world semantics also properly treats both types of disjunctions in a general disjunctive program and provides the most careful negative inference compared with the GCWA$^{\neg}$ and the WGCWA$^{\neg}$. Moreover, when a program is inconsistent, the PWA$^{\neg}$ often behaves interestingly.

Example 3.4 Let $P = \{a \vee b \leftarrow,\ \ b \leftarrow a,\ \ \leftarrow not\, a,\ \ c \leftarrow not\, b\}$. Then $\mathcal{ST}_P = \emptyset$, $\mathcal{ST}_{NP(P)} = \{\{a,b\}\}$, and $\mathcal{PW}_P = \{\{a,b\}\}$, hence $GCWA^{\neg}(P)$ is not well-defined, while $PWA^{\neg}(P)$ and $WGCWA^{\neg}(P)$ imply $\neg c$. □

Note that the above program is inconsistent (hence Lemma 3.3 does not hold here), but $\{a,b\}$ is a model of P (Lemma 3.2). Observing the above program, the third clause asserts that a should be true, which possibly holds by the first disjunctive clause. While, the truth of a implies the truth of b in the second clause, then it seems natural to assert the falsity of c by the last clause.

By definition, the PWA$^{\neg}$ is well-defined whenever the GCWA$^{\neg}$ or the WGCWA$^{\neg}$ is. Further, the above example suggests that even in an inconsistent program, the PWA$^{\neg}$ often infers proper negation which cannot be obtained under the GCWA$^{\neg}$.

4 Computing Negation

4.1 Bottom-up Model Generation Proof Procedure

The algorithm we use to compute negation in disjunctive programs is based upon the bottom-up model generation proof procedure. In this section, we consider a program which consists of clauses of the form:

$$\Gamma_1 \vee \ldots \vee \Gamma_l \leftarrow B_1 \wedge \ldots \wedge B_m$$

where Γ_i $(1 \le i \le l)$ is a conjunction of atoms, B_j $(1 \le j \le m)$ is an atom, and all variables are assumed to be universally quantified at the front of the clause. A positive disjunctive program is regarded as a special case where each Γ_i is an atom. We also assume here and in the next subsection that a program is function-free and range-restricted[2], such conditions are usually imposed upon a program in the context of deductive databases.

Let P be a program presented above and $conj(\Gamma_j)$ be the set of conjuncts from Γ_j. Then for a given set of interpretations $\mathcal{I}_P^i$, the following algorithm generates the new set of interpretations $\mathcal{I}_P^{i+1}$. Let $\mathcal{I}_P^0 = \{\emptyset\}$ and $NOGOOD = \emptyset$. For $i \ge 0$ do:

1. For every non-negative clause C_k in P of the form:

   $$C_k : \Gamma_1 \vee \ldots \vee \Gamma_l \leftarrow B_1 \wedge \ldots \wedge B_m$$

   such that $I \in \mathcal{I}_P^i$ and $I \models (B_1 \wedge \ldots \wedge B_m)\sigma$ for some substitution σ, put $I \cup \bigcup_{C_k} \{conj(\Gamma_j)\sigma\}$
   $(1 \le j \le l)$ into $\mathcal{I}_P^{i+1}$ if it is not a superset of any element of $NOGOOD$.

2. If there is a negative clause in P of the form:

   $$\leftarrow B_1 \wedge \ldots \wedge B_m$$

   such that $I \in \mathcal{I}_P^i$ and $I \models (B_1 \wedge \ldots \wedge B_m)\sigma$ for some substitution σ, then put I into $NOGOOD$.

3. Iterate the above two steps until it reaches the fixpoint $\mathcal{I}_P^{n+1} = \mathcal{I}_P^n$ which is closed under the above two operations.

The above procedure performs forward reasoning based upon hyper-resolution and case-splitting on non-unit derived clauses. Note here that since a program is range-restricted, each disjunct Γ_j generated in step 1 is completely instantiated, hence soundness of case-splitting is guaranteed [MB88]. Moreover, since we consider a finite function-free program, the above procedure always terminates in a finite step. The $NOGOOD$ records unsatisfiable interpretations of a program, which is used to avoid unnecessary expansion during the closure computation.

The next theorem presents that the fixpoint closure computed by the above procedure exactly provides the set of all possible worlds of a program. In the following, let $\mathcal{I}_P^\omega$ be the fixpoint closure obtained by the above procedure.

Theorem 4.1 Let P be a positive disjunctive program and $\mathcal{PW}_P$ be the set of all possible worlds of P. Then $\mathcal{PW}_P = \mathcal{I}_P^\omega$.

[2]That is, any variable in a clause has its occurrence in a positive atom in the body.

Proof: I is in $\mathcal{I}_P^{\omega}$ iff for each A_i in I, there is a ground clause $A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m$ from P such that $1 \leq i \leq l$ and $\{B_1, \ldots, B_m\} \subseteq I$
iff I is the least Herbrand model of a split program P' in **Horn**(P) such that $A_i \leftarrow B_1 \wedge \ldots \wedge B_m$ is in P'
iff I is in $\mathcal{PW}_P$. □

Especially, if $\mathcal{I}_P^{\omega} = \emptyset$, P is inconsistent. By Lemma 2.4, the following result directly follows.

Corollary 4.2 Let P be a positive disjunctive program and $\mathcal{MM}_P$ be the set of all minimal models of P. Then $\mathcal{MM}_P = min(\mathcal{I}_P^{\omega})$ where $min(\mathcal{I}_P^{\omega}) = \{I \in \mathcal{I}_P^{\omega} \mid \nexists J \in \mathcal{I}_P^{\omega} \text{ such that } J \subset I\}$. □

Theorem 4.3 For a consistent positive disjunctive program P and a ground atom A,

(i) $GCWA(P) \models \neg A$ iff $A \notin I$ for any $I \in min(\mathcal{I}_P^{\omega})$.
(ii) If $P \cup Horn(P)$ is consistent, then $WGCWA(P) \models \neg A$ iff $A \notin I$ for any $I \in \mathcal{I}_P^{\omega}$.
(iii) $PWA(P) \models \neg A$ iff $A \notin I$ for any $I \in \mathcal{I}_P^{\omega}$.

Proof: (i) and (iii) directly follow from each definition and the above theorem/corollary. When $P \cup Horn(P)$ is consistent, the WGCWA coincides with the PWA (Theorem 2.6), hence the result also holds. □

Example 4.1 Let $P = \{a \vee b \leftarrow, \ \leftarrow b\}$ where $P \cup Horn(P)$ is inconsistent. Then $b \notin \{a\} \in \mathcal{I}_P^{\omega}$, while $WGCWA(P) \not\models \neg b$. □

4.2 Program Transformation

For general disjunctive programs, Inoue et al [IKH92] have proposed a program transformation which transforms a general disjunctive program into a semantically equivalent *not*-free program. According to [IKH92], given a general disjunctive program P, each clause

$$A_1 \vee \ldots \vee A_l \leftarrow B_1 \wedge \ldots \wedge B_m \wedge not B_{m+1} \wedge \ldots \wedge not B_n$$

in P is transformed into the following clause in P^{κ}:

$$\begin{aligned}(A_1 \wedge \neg \mathsf{K} B_{m+1} \wedge \ldots \wedge \neg \mathsf{K} B_n) \vee \ldots \vee (A_l \wedge \neg \mathsf{K} B_{m+1} \wedge \ldots \wedge \neg \mathsf{K} B_n) \\ \vee \mathsf{K} B_{m+1} \vee \ldots \vee \mathsf{K} B_n \leftarrow B_1 \wedge \ldots \wedge B_m \end{aligned} \qquad (1)$$

In addition, for each atom A from P, the following clauses are in P^{κ}:

$$\leftarrow A \wedge \neg \mathsf{K}A \tag{2}$$

$$\leftarrow \mathsf{K}A \wedge \neg \mathsf{K}A \tag{3}$$

where $\mathsf{K}A$ and $\neg\mathsf{K}A$ are newly introduced *atoms* meaning A *is believed* and *disbelieved*, respectively.

In the above transformation, each *not* B_j in the body is rewritten in $\neg\mathsf{K}B_j$ and shifted to the head of the clause. An intuitive reading of each rule is that (1) if $B_1, \ldots, B_m$ are true, then some A_i $(1 \leq i \leq l)$ becomes true with the condition that $B_{m+1}, \ldots, B_n$ are disbelieved; otherwise, some B_j $(m+1 \leq j \leq n)$ is believed. While, (2) (resp. (3)) says that it cannot happen that A is true (resp. believed) and disbelieved at the same time.

Using this translation, every general disjunctive program P is transformed into a *not*-free disjunctive program P^{κ}. Since P^{κ} is a subclass of the program presented in the previous section, its model generation proof procedure is already defined. Let I^{κ} be an interpretation of P^{κ}. Then I^{κ} is called *canonical*, if $\mathsf{K}A \in I^{\kappa}$ implies $A \in I^{\kappa}$ for each atom A. Given the interpretation I^{κ} and the set of interpretations $\mathcal{I}_{P^{\kappa}}$, let $obj(I^{\kappa}) = I^{\kappa} \cap \mathcal{HB}_P$ and $obj_c(\mathcal{I}_{P^{\kappa}}) = \{obj(I^{\kappa}) \mid I^{\kappa} \in \mathcal{I}_{P^{\kappa}} \text{ and } I^{\kappa} \text{ is canonical}\}$. Then the following relationship holds.

Theorem 4.4 [IKH92] [3] Let P be a general disjunctive program and P^{κ} be its transformed program. Then $\mathcal{ST}_P = obj_c(min(\mathcal{I}^{\omega}_{P^{\kappa}}))$. Especially, if $obj_c(min(\mathcal{I}^{\omega}_{P^{\kappa}})) = \emptyset$, P is inconsistent. □

Lemma 4.5 [IKH92, IS92] Let P be a general logic program and P^{κ} be its transformed program. Then $\mathcal{ST}_P = obj_c(\mathcal{I}^{\omega}_{P^{\kappa}})$. □

Theorem 4.6 Let P be a general disjunctive program. Then $\mathcal{PW}_P = obj_c(\mathcal{I}^{\omega}_{P^{\kappa}})$

Proof: Let I be a stable model of some consistent general logic program P' in $\mathbf{GLP}(P)$. Then by Lemma 4.5, I is in $obj_c(\mathcal{I}^{\omega}_{P'^{\kappa}})$. Since P' is a program obtained by splitting each disjunctive clause in P, $\mathcal{I}^{\omega}_{P'^{\kappa}}$ is a subset of $\mathcal{I}^{\omega}_{P^{\kappa}}$ Hence I is also in $obj_c(\mathcal{I}^{\omega}_{P^{\kappa}})$. The converse is also shown in the same manner. □

Theorem 4.7 For a consistent general disjunctive program P and a ground atom A,

(i) $GCWA^{\neg}(P) \models \neg A$ iff $A \notin I$ for any $I \in obj_c(min(\mathcal{I}^{\omega}_{P^{\kappa}}))$.

[3] In [IKH92], a slightly different procedure is used, but the result still holds here.

(ii) $WGCWA^{\neg}(P) \models \neg A$ iff $A \notin I$ for any $I \in obj_c(min(\mathcal{I}^{\omega}_{P^{\kappa}})) \cup obj_c(\mathcal{I}^{\omega}_{NP(P)^{\kappa}})$.

(iii) $PWA^{\neg}(P) \models \neg A$ iff $A \notin I$ for any $I \in obj_c(\mathcal{I}^{\omega}_{P^{\kappa}})$.

Proof: (i) and (iii) directly follow from Theorem 4.4 and 4.6. (ii) also follows from Lemma 4.5 and the definition of the WGCWA$^{\neg}$. $\square$

Example 4.2 (cont. from Example 3.4) The program P is transformed into $P^{\kappa} = \{a \vee b \leftarrow, \; b \leftarrow a, \; \mathsf{K}a \leftarrow, \; (c \wedge \neg \mathsf{K}b) \vee \mathsf{K}b \leftarrow\} \cup \{\leftarrow A \wedge \neg \mathsf{K}A, \; \leftarrow \mathsf{K}A \wedge \neg \mathsf{K}A \mid A = a, b, c\}$. Then $\mathcal{I}^{\omega}_{P^{\kappa}} = \{\{a, b, \mathsf{K}a, \mathsf{K}b\}, \{b, \mathsf{K}a, \mathsf{K}b\}\}$. Thus, $obj_c(\mathcal{I}^{\omega}_{P^{\kappa}}) = \{\{a, b\}\}$, which contains the unique possible world of P. On the other hand, $min(\mathcal{I}^{\omega}_{P^{\kappa}}) = \{\{b, \mathsf{K}a, \mathsf{K}b\}\}$, then $obj_c(min(\mathcal{I}^{\omega}_{P^{\kappa}})) = \emptyset$, hence P has no stable model. $\square$

5 Related Work

The GCWA and the WGCWA are initially introduced for positive disjunctive programs in [Min82, RLM89]. For stratified disjunctive programs, the GCWAS [RM89] and the ICWA [GPP89] are known as the extensions of the GCWA. The GCWA$^{\neg}$ clearly reduces to them in stratified disjunctive programs. The GCWA$^{\neg}$ is a direct extension of the GCWA and is usually assumed to infer negation under the stable model semantics when a program has multiple stable models. Alternative approaches for inferring negation in general disjunctive programs are presented by several researchers in the context of the *extended well-founded semantics* [Ros89, BLM90, Prz90b, Prz91, Dix92]. Under the well-founded semantics, negation assumed under the CWA corresponds to the *unfounded set* [VRS91] of a program. Although all of these approaches are the extensions of the well-founded semantics, each semantics provides a slightly different framework with each other. (A comparison between some of them is presented in [BLM90, Dix92].) Roughly speaking, the difference between the GCWA$^{\neg}$, the WGCWA$^{\neg}$ and those previously proposed approaches corresponds to the difference between the stable model semantics and the well-founded semantics of normal programs.

The possible world semantics is also independently discovered by Chan [Cha89] and lately by Decker [Dec92] under the name of the *supported model semantics*. Decker and Casamayor [DC92] have also shown that their *supported world assumption*, which corresponds to the PWA, satisfies the properties such as *cautious monotonicity*, *cumulativity* and *rationality* in the sense of [KLM90]. While these works have characterized the PWS from different points of view, they consider only positive disjunctive programs and its extension to general disjunctive programs is not studied in the literature. Ross [Ros89] has proposed the *optimal* well-founded semantics which can treat both inclusive and exclusive disjunctions in a general disjunctive program. However, his semantics requires each rule to be *clarified* whether it is exclusive or inclusive, and it cannot treat a disjunctive clause containing both types of disjunctions at the same time as is presented in the introductory example. Recently, Dung [Dun91] has also presented a *completion* theory of negation which can distinguish both types of disjunctions in a program. However, it is defined for only positive disjunctive programs and also cannot treat both types of disjunctions in the same clause. Gelfond [Gel91] has developed another theory of negation from the epistemic point of view, which is also different from ours.

To distinguish two kinds of disjunctions, one may consider that instead of inserting negative clauses, inserting *cyclic* clauses under the usual minimal model semantics is enough. But this is not the case. Consider to interpret the disjunction $a \vee b$ inclusively, adding cyclic clauses $a \leftarrow b$ and $b \leftarrow a$ to it. The resultant program now implies the equivalence $a \Leftrightarrow b$. Applying it to the introductory example, it implies $Sunday \Leftrightarrow National\text{-}holiday$, which is of course not our intention.

We have used negative clauses to distinguish exclusive disjunctions from inclusive ones. However, instead of using negative clauses, we can also use *explicit negation* in the context of *extended logic programs* [GL91]. For instance, we can replace $\leftarrow a \wedge b$ by $\neg a \vee \neg b \leftarrow$ in an *extended disjunctive program* and also introduce the possible world semantics for extended disjunctive programs which is defined in the same manner presented in this paper. Note here that the *answer set semantics* [GL91] of extended disjunctive programs is also based upon the minimal model semantics, hence cannot distinguish two kinds of disjunctions in general. For example, the programs $\{a \vee b \leftarrow\}$ and $\{a \vee b \leftarrow,\ \ \neg a \vee \neg b \leftarrow\}$ have answer sets $\{a\}$, $\{b\}$ and $\{a, \neg b\}$, $\{\neg a, b\}$, respectively. In both programs, the disjunction is treated exclusively.

A bottom-up model generation proof procedure for computing minimal and stable models in disjunctive programs is also developed in [FM91, FLMS91]. Compared with theirs, our algorithm is dedicated for computing not only stable models, but also possible worlds. Further, our algorithm has some computational advantages over them even for computing stable models [IS92]. Chan [Cha89] also presents a different procedure which, given a positive disjunctive program P and its model M, finds a subset of M that is also a possible world of P. A top-down proof procedure for evaluating queries under the possible world semantics is also presented in [Sak89].

6 Concluding Remarks

This paper has presented a theory of negation for disjunctive logic programs. The GCWA, the WGCWA and the PWA for positive disjunctive programs are naturally extended to the $GCWA^{\neg}$, the $WGCWA^{\neg}$ and the $PWA^{\neg}$, respectively for general disjunctive programs. It is shown that the $GCWA^{\neg}$ is stronger than the $WGCWA^{\neg}$. On the other hand, the $PWA^{\neg}$ provides the most careful negative inference compared with the other two, and often infers proper negation from an inconsistent general disjunctive program. We have also presented a bottom-up model generation proof procedure for computing possible worlds and negation in disjunctive logic programs. It is sound and complete to compute stable models, possible worlds, and corresponding each negation for function-free range-restricted programs. The proof procedure is also implemented on a bottom-up parallel model generation theorem prover called $MGTP$ developed at ICOT.

The possible world semantics presented in this paper is based upon the stable model semantics of general logic programs, hence it does not satisfy *cumulativity* nor *modularity* principle in general [Dix92]. However, these properties are not serious shortcomings of the possible world semantics; if one desires such properties, we can easily construct an alternative possible world semantics based upon another cumulative and modular semantics such as the well-founded semantics. In fact, our possible world semantics is defined through the set of general logic programs, it is easy to construct its well-founded version by employing the well-founded models instead of stable models in its definition. In other words, *we can construct a possible world semantics of disjunctive programs corresponding to any semantics for general logic programs.* And such a possible world semantics promises to have nice properties as is presented in this paper.

References

[BLM90] Baral, C., Lobo, J. and Minker, J., Generalized Disjunctive Well-Founded Semantics for Logic Programs, Research Report CS-TR-2436, Dept. of Computer Science, Univ. of Maryland, 1990.

[Cha89] Chan, E. P. F., A Possible World Semantics for Non-Horn Databases, Research Report CS-89-47, Dept. of Computer Science, Univ. of Waterloo, 1989.

[Cla78] Clark, K. L., Negation as Failure, in *Logic and Data Bases* (H. Gallaire and J. Minker eds.), Plenum, New York, 293-322, 1978.

[Dec92] Decker, H., Foundations of First-Order Databases, Research Report, Siemens, 1992.

[DC92] Decker, H. and Casamayor, J. C., Supported Models and Supported Answers in First Order Databases, Draft Manuscript, 1992.

[Dun91] Dung, P. M., Negation as Failure for Disjunctive Logic Programming, *Proc. ILPS Workshop on Disjunctive Logic Programs*, 1991.

[Dix92] Dix, J., Classifying Semantics of Disjunctive Logic Programs, *Proc. Joint Int. Conf. and Symp. on Logic Programming*, Washington, 798-812, 1992.

[FM91] Fernandez, J. A. and Minker, J., Computing Perfect Models of Disjunctive Stratified Databases, *Proc. ILPS Workshop on Disjunctive Logic Programs*, 1991.

[FLMS91] Fernandez, J. A., Lobo, J., Minker, J. and Subrahmanian, V. S., Disjunctive LP + Integrity Constraints = Stable Model Semantics, *Proc. ILPS Workshop on Deductive Databases*, 110-117, 1991.

[GL88] Gelfond, M. and Lifschitz, V., The Stable Model Semantics for Logic Programming, *Proc. 5th Int. Conf. Symp. on Logic Programming*, 1070-1080, 1988.

[GL91] Gelfond, M. and Lifschitz, V., Classical negation in logic programs and disjunctive databases, *New Generation Computing* 9, 365-385, 1991.

[Gel91] Gelfond, M., Epistemic Semantics for Disjunctive Databases, *Proc. ILPS Workshop on Disjunctive Logic Programs*, 1991.

[GPP89] Gelfond, M., Przymusinska, H. and Przymusinski, T., On the Relationship between Circumscription and Negation as Failure, *Artificial Intelligence* 38, 75-94, 1989.

[IKH92] Inoue, K., Koshimura, M. and Hasegawa, R., Embedding Negation as Failure into a Model Generation Theorem Prover, *Proc. 11th Int. Conf. on Automated Deduction, Lecture Notes in Artificial Intelligence 607*, Springer-Verlag, 400-415, 1992.

[IS92] Inoue, K. and Sakama, C., A Uniform Approach to Fixpoint Characterization of Disjunctive and General Logic Programs, ICOT Technical Report TR-817, 1992.

[KLM90] Kraus, S., Lehmann, D. and Magidor, M., Nonmonotonic Reasoning, Preferential Models and Cumulative Logics, *Artificial Intelligence* 44(1), 167-207, 1990.

[LMR92] Lobo, J., Minker, J. and Rajasekar, A., Foundations of Disjunctive Logic Programming, MIT Press, 1992.

[MB88] Manthey, R. and Bry, F., SATCHMO: a theorem prover implemented in Prolog, *Proc. 9th Int. Conf. on Automated Deduction*, Lecture Notes in Computer Science 310, 415–434, Springer-Verlag, 1988.

[Min82] Minker, J., On Indefinite Data Bases and the Closed World Assumption, *Proc. 6th Int. Conf. on Automated Deduction, Lecture Notes in Computer Science 138*, Springer-Verlag, 292-308, 1982.

[Prz90a] Przymusinski, T. C., Extended Stable Semantics for Normal and Disjunctive Logic Programs, *Proc. 7th Int. Conf. on Logic Programming*, Jerusalem, 459-477, 1990.

[Prz90b] Przymusinski, T. C., Stationary Semantics for Disjunctive Logic Programs and Deductive Databases, *Proc. North American Conf. on Logic Programming*, Austin, 40-62, 1990.

[Prz91] Przymusinski, T. C., Semantics of Disjunctive Logic Programs and Deductive Databases, *Proc. 2nd Int. Conf. on Deductive and Object-Oriented Databases*, Munich, 85-107, 1991.

[Rei78] Reiter, R., On Closed World Databases, in *Logic and Data Bases* (H. Gallaire and J. Minker eds.), Plenum, New York, 55-76, 1978.

[RLM89] Rajasekar, A., Lobo, J. and Minker, J., Weak Generalized Closed World Assumption, *J. Automated Reasoning* 5, 293-307, 1989.

[Ros89] Ross, K., The Well Founded Semantics for Disjunctive Logic Programs, *Proc. 1st Int. Conf. on Deductive and Object-Oriented Databases*, Kyoto, 352-369, 1989.

[RM89] Rajasekar, A. and Minker, J., A Stratification Semantics for General Disjunctive Programs, *Proc. North American Conf. on Logic Programming*, 573-586, 1989.

[RT88] Ross, K. A. and Topor, R. W., Inferring Negative Information from Disjunctive Databases, *J. Automated Reasoning* 4, 397-424, 1988.

[Sak89] Sakama, C., Possible Model Semantics for Disjunctive Databases, *Proc. 1st Int. Conf. on Deductive and Object-Oriented Databases*, Kyoto, 337-351, 1989.

[VRS91] Van Gelder, A., Ross, K. and Schlipf, J. S., The Well-Founded Semantics for General Logic Programs, *J. ACM* 38(3), 620-650, 1991.