TM-0692

Possible Model Semantics for Disjunctive Databases

by

C. Sakama & H. Seki

March, 1989

Possible Model Semantics for Disjunctive Databases (Preliminary Report)

Chiaki Sakama and Hirohisa Seki

ICOT

1-4-28, Mita, Minato-ku, Tokyo 108, Japan

Abstract

This paper presents a declarative semantics of disjunctive databases based on the *possible model semantics*. The possible model semantics is an extension of the minimal model semantics of databases and provides a natural meaning of disjunctive databases. We characterize it by giving a new fixpoint semantics of databases and present the *generalized closed world assumption for the possible model semantics ($GCWA_P$)* for inferring negative information from a database. We also discuss some extension and its relation to previous approaches.

1 Introduction

The declarative semantics of deductive databases has been widely studied based on the minimal model semantics by incorporating the inference rule for negation. For definite Horn databases, the unique *least Herbrand model* gives the declarative meaning of a database [VK76] and the *closed world assumption (CWA)* [Re78] provides negative information as the facts which are not true in this model.

When a database is indefinite and contains some non-Horn clauses, there is usually no unique minimal model, and some *preferred* minimal models are often selected as the intended models of the database. For example, *standard models* [ABW88] or *perfect models* [Pr88] are introduced for stratified databases as intended minimal models of a database, and correspondingly, the *iterated CWA (ICWA)* [GPP86b] provides negative information as the complement of such a model. These results are extended to the *extended CWA(ECWA)*[GPP86a,GPP86b] and equivalently, *circumscription* [Mc86].

By contrast, the so-called *disjunctive database* is an indefinite database which usually has no minimal model that prefers to others. Then, there are two alternative approaches for such databases. One considers the collection of minimal models and the facts or sentences which are not true in any minimal model are assumed to be false by the *generalized CWA (GCWA)* [Mi82] and the *extended GCWA (EGCWA)* [YH85]. The other approaches which are *not* based on the minimal model semantics are the *disjunctive database rule (DDR)* [RT88] and the *weak GCWA (WGCWA)* [LMR88].

Now we compare two alternative rules, the GCWA and the DDR, for disjunctive databases. Suppose an example, $'study(Jack, Math) \vee study(Jack, Physics)'$ (*Jack studies math or physics*). Then if we know $'study(Jack, Math)'$ is true, then $' \sim study(Jack, Physics)'$ is inferred by the GCWA. (From the EGCWA, $' \sim (study(Jack, Math) \wedge study(Jack, Physics))'$ is inferred without knowing $'study(Jack, Math)'$.) That is, the GCWA does not allow the possibility of Jack's studying both. The DDR, on the other hand, does not infer $' \sim study(Jack, Physics)'$ nor $' \sim (study(Jack, Math) \wedge study(Jack, Physics))'$, and seems to be more flexible than the GCWA. Suppose an another example, $'marry(Mary, John) \vee marry(Mary, Dick)'$ (*Mary will marry John or Dick*). In this situation, however, it seems natural that knowing $'marry(Mary, John)'$ is true implies that $'marry(Mary, Dick)'$ is false by the GCWA.

The problem is that the DDR is too weak since it interprets each disjunction in an *inclusive* way, while the GCWA is too strong since it interprets each disjunction in an *exclusive* way. To distinguish an exclusive disjunction from an inclusive one, it is sufficient to use some negative clauses to present exclusive disjunctions (this is also suggested in [RT88], section 9). In the above example, by adding a clause $' \leftarrow marry(Mary, John) \wedge marry(Mary, Dick)'$, we get the intended meaning of the sentences.

This paper presents a declarative semantics of disjunctive databases which can handle both inclusive and exclusive disjunctions using negative clauses. In section 2, we present a declarative semantics of disjunctive databases based on the class of *possible model*, which is an extension of minimal model semantics and provides an intended meaning of disjunctive databases. Then we give a new fixpoint semantics of databases in the presence of negative clauses, using a continuous mapping from a powerset of the Herbrand interpretation to itself, and show that its least fixed point is a set of the possible models of a database. We also present the *generally closed world assumption for possible model semantics (GCWA$_P$)* for inferring negative information from a database which provides a reasonable compromise between the GCWA and the DDR. In section 3, we extend the results to stratified disjunctive databases and discuss the relation to previous approaches.

2 Possible Model Semantics for Disjunctive Databases

2.1 Possible Model

First, we give the class of databases considered in this section.

Definition A *database* is a finite *consistent* set of the clauses of the form:

$A_1 \vee ... \vee A_m \leftarrow B_1 \wedge ... \wedge B_n$

where $m, n \geq 0$ and each A_i and B_j are atoms, and all variables are universally quantified at the front of the clause. $A_1 \vee ... \vee A_m$ is called the *head* of the clause, and $B_1 \wedge ... \wedge B_n$ is called the *body* of the clause. When $m > 1$, we call the clause *disjunctive*. □

The definition assumes a database to be consistent, but in general, a set of clauses which contains negative clauses has the possibility of being inconsistent. When it is considered a set of Horn clauses, we can check the consistency of the set by taking each negative clause as a goal for the set of positive clauses and examining whether it has a refutation or not. If any of these goals has a refutation, then the set is inconsistent [SM83]. From another point of view, negative clauses are considered as integrity constraints for the database, and the above procedure corresponds to that of checking the satisfiability of integrity constraints [LI87].

Such a procedure may be extended to disjunctive databases by using some query evaluation procedure, but it is beyond the scope of this paper.

Next we introduce the notion of the *split database.*[1]

Definition Let C be a disjunctive clause in D of the form:

$A_1 \vee ... \vee A_m \leftarrow B_1 \wedge ... \wedge B_n$

Then C is *split* into $2^m - 1$ sets of clauses $\mathbf{C}_1, .., \mathbf{C}_{2^m-1}$ where

$\mathbf{C}_j = \{A_i \leftarrow B_1 \wedge ... \wedge B_n \mid A_i \in S \text{ where } S \in 2^{\{A_1,...,A_m\}} \setminus \emptyset\}$ $(1 \leq j \leq 2^m - 1)$

The *split database* of D is a database where each disjunctive clause C is replaced by the split clauses $\mathbf{C}_j$. □

If a database contains k disjunctive clauses where each has m_i atoms in its head, then the database is split into *at most* $\prod_{i=1}^{k}(2^{m_i} - 1)$ different databases.

Example Let $D = \{P \vee Q \leftarrow R,\ R \vee S \leftarrow T,\ T,\ \leftarrow P \wedge Q\}$. Then it is split into the following seven databases.

$D_1 = \{P \leftarrow R,\ R \leftarrow T,\ T,\ \leftarrow P \wedge Q\}$. $D_2 = \{P \leftarrow R,\ S \leftarrow T,\ T,\ \leftarrow P \wedge Q\}$.

$D_3 = \{P \leftarrow R,\ R \leftarrow T,\ S \leftarrow T,\ T,\ \leftarrow P \wedge Q\}$. $D_4 = \{Q \leftarrow R,\ R \leftarrow T,\ T,\ \leftarrow P \wedge Q\}$.

[1]The notion of split database is also introduced in [Lo78] in a different context.

$D_5 = \{Q \leftarrow R,\ \ S \leftarrow T,\ \ T,\ \ \leftarrow P \wedge Q\}$. $D_6 = \{Q \leftarrow R,\ \ R \leftarrow T,\ \ S \leftarrow T,\ \ T,\ \ \leftarrow P \wedge Q\}$.

$D_7 = \{P \leftarrow R,\ \ Q \leftarrow R,\ \ S \leftarrow T,\ \ T,\ \ \leftarrow P \wedge Q\}$.

The following two are, however, inconsistent and not databases.

$D_8 = \{P \leftarrow R,\ \ Q \leftarrow R,\ \ R \leftarrow T,\ \ T,\ \ \leftarrow P \wedge Q\}$.

$D_9 = \{P \leftarrow R,\ \ Q \leftarrow R,\ \ R \leftarrow T,\ \ S \leftarrow T,\ \ T,\ \ \leftarrow P \wedge Q\}$. □

Now we define the notion of possible model.

Definition Let D be a database and M be an Herbrand model of D. Then M is called *possible* if it is a minimal model of a split database of D. □

Example Let D be the same database as the previous example. Then $\{P, R, T\}, \{Q, R, T\}$, $\{S, T\}, \{P, R, S, T\}$ and $\{Q, R, S, T\}$ are the possible models of D, while $\{P, S, T\}$ and $\{Q, S, T\}$ are models of D but not possible. □

In the following, 'model' means an Herbrand model.

An order relation $\preceq$ between possible models is defined such that for each possible model M and N, $M \preceq N$ iff $M \subseteq N$. It is clear that a set of possible models of a database makes a complete lattice under the $\preceq$.

2.2 Fixpoint Semantics

In this section, we give a new fixpoint semantics of a database. First, we define a mapping T from an Herbrand interpretation to a powerset of Herbrand interpretation [SS89].

Definition Let D be a database and B be the Herbrand base of D. The mapping $T_D : 2^B \rightarrow 2^{2^B}$ is defined as follows. Let I be an Herbrand interpretation. Then $T_D(I) = \{I' \in 2^B \mid A_1 \vee \ldots \vee A_m \leftarrow B_1 \wedge \ldots \wedge B_n$ is a ground instance of a clause C_i in D. If $I \models B_1 \wedge \ldots \wedge B_n$ and $m = 0$ for some i then $I' = \emptyset$. Else, if $I \models B_1 \wedge \ldots \wedge B_n$ and $m \neq 0$ for all i then $I' = I \bigoplus \bigotimes_{C_i \in D}(2^{\{A_1,\ldots,A_m\}} \setminus \emptyset)\}$.

In the above, $\bigotimes_{C_i \in D}(2^{\{A_1,\ldots,A_m\}} \setminus \emptyset) = \{J \mid J \in \bigcup_i J^i$ where $J^i \in 2^{\{A_1,\ldots,A_m\}} \setminus \emptyset\}$ and $I \bigoplus \mathbf{J} = \{I \cup J \mid J \in \mathbf{J}\}$. □

Example Consider the database $D = \{P(X) \vee Q(X) \leftarrow R(X),\ \ S(f(X)) \leftarrow R(X),$ $\leftarrow R(g(X)),\ \ R(a)\}$. Then $T_D(\{R(a)\}) = \{\{R(a), S(f(a)), P(a)\}, \{R(a), S(f(a)), Q(a)\},$ $\{R(a), S(f(a)), P(a), Q(a)\}\}$ and $T_D(\{R(g(a))\}) = \{\emptyset\}$. □

We then define a mapping from a powerset of the Herbrand interpretation to itself.

Definition Let D be a database. The mapping $\mathbf{T_D} : 2^{2^B} \rightarrow 2^{2^B}$ is defined as follows.

$\mathbf{T_D}(\mathbf{I}) = \bigcup_{J \in \mathbf{I}} T_D(J)$

where $\mathbf{I}$ is a set of Herbrand interpretations. □

Example Consider the database in the previous example. Then $\mathbf{T_D}(\{\{R(a)\}, \{R(f(a))\}\}) = T_D(\{R(a)\}) \cup T_D(\{R(f(a))\})$. □

Clearly, 2^{2^B} makes a complete lattice under set inclusion. In fact, the least upper bound of a collection of subsets of 2^B is defined by their union and the greatest lower bound is defined by their intersection. The top element is 2^B, and the bottom element is $\{\emptyset\}$.

Lemma 2.2.1 The mapping $\mathbf{T_D}$ is monotonic.

Proof From the definition of $\mathbf{T_D}$, it is clear that $\mathbf{I} \subseteq \mathbf{J}$ implies $\mathbf{T_D}(\mathbf{I}) \subseteq \mathbf{T_D}(\mathbf{J})$ where $\mathbf{I}$ and $\mathbf{J}$ are the sets of Herbrand interpretations. □

Note that T_D is not monotonic. For example, let $D = \{P \leftarrow Q, \ \leftarrow R\}$ then $T_D(\{Q\}) = \{\{Q, P\}\}$ and $T_D(\{Q, R\}) = \{\emptyset\}$. Thus, $\{Q\} \subset \{Q, R\}$ does not imply $T_D(\{Q\}) \subseteq T_D(\{Q, R\})$.

Corollary 2.2.2 Let D be a database and $\mathbf{X}$ be a directed subset of 2^{2^B}. Then $I \in lub(\mathbf{X})$ iff $\exists \mathbf{I} \in \mathbf{X}$ and $I \subset \mathbf{I}$

Proof ($\rightarrow$) It suffices to put $\mathbf{I} = lub(\mathbf{X}) \in \mathbf{X}$.

($\leftarrow$) It is clear since $I \in \mathbf{I} \subseteq lub(\mathbf{X})$. □

Lemma 2.2.3 The mapping $\mathbf{T_D}$ is continuous.

Proof We show $\mathbf{T_D}(lub(\mathbf{X})) = lub(\mathbf{T_D}(\mathbf{X}))$ where $\mathbf{X}$ is a directed subset of 2^{2^B}.

$I \in \mathbf{T_D}(lub(\mathbf{X}))$

iff $\exists J \in lub(\mathbf{X})$ and $I \in T_D(J)$

iff $\exists \mathbf{I} \subset \mathbf{X}$, $J \in \mathbf{I}$ and $I \in T_D(J)$ (by corollary 2.2.2)

iff $\exists \mathbf{I} \in \mathbf{X}$ and $I \in \mathbf{T_D}(\mathbf{I})$

iff $I \in lub(\mathbf{T_D}(\mathbf{X}))$ □

Lemma 2.2.4 [SS89] Let D be a database and I be an Herbrand interpretation of D. Then I is a model of D iff $I \in \mathbf{T_D}(\{I\})$.

Proof I is a model of D

iff for each ground instance $A_1 \vee ... \vee A_m \leftarrow B_1 \wedge ... \wedge B_n$, of each clause in D, if $m = 0$ then $I \not\models B_1 \wedge ... \wedge B_n$, else $I \models B_1 \wedge ... \wedge B_n$ implies $I \models A_i$ for some $A_i (1 \leq i \leq m)$

iff $I \in T_D(I)$ (by the definition of T_D)

iff $I \in \mathbf{T_D}(\{I\})$ □

Definition Let D be a database and $\mathbf{I}$ be a set of Herbrand interpretations of D. Then the ordinal powers of $\mathbf{T_D}$ are defined as $\mathbf{T_D} \uparrow 0(\mathbf{I}) = \mathbf{I}$, $\mathbf{T_D} \uparrow (n+1)(\mathbf{I}) = \mathbf{T_D}(\mathbf{T_D} \uparrow n(\mathbf{I}))$, and $\mathbf{T_D} \uparrow \omega(\mathbf{I}) = \{I \mid \exists \alpha < \omega,\ \ I \in \bigcap_{n \geq \alpha} \mathbf{T_D} \uparrow n(\mathbf{I})\}$, where n is a successor ordinal and ω is a limit ordinal. □

Lemma 2.2.5 Let D be a database. Then $lfp(\mathbf{T_D}) = \mathbf{T_D} \uparrow \omega(\{\emptyset\})$.

Proof The result follows from lemma 2.2.3 and proposition 5.4 in [Ll87]. □

Definition Let D be a database. Then the *fixpoint semantics* of D, FP_D, is defined as:
$FP_D = \{I \mid I \in lfp(\mathbf{T_D})$ and $I \in \mathbf{T_D}(\{I\})\}$. □

The following theorem gives a fixpoint characterisation of the possible model semantics of a database.

Theorem 2.2.6 Let D be a database. Then I is in FP_D iff I is a possible model of D.

Proof $I \in FP_D$

iff $I \in \mathbf{T_D}(\{I\})\}$ and $\exists n < \omega$ s.t. $I \in \mathbf{T_D} \uparrow n(\{\emptyset\})$ (by lemma 2.2.5)

iff I is a model of D (by lemma 2.2.4) and for each A in I, there is a ground instance

$A_1 \vee \ldots \vee A_m \leftarrow B_1 \wedge \ldots \wedge B_n$

of a clause in D such that $A = A_i (1 \leq i \leq m)$ and $I \models B_1 \wedge \ldots \wedge B_n$

iff I is a model of some split database D' of D and for each A in I, D' includes a ground instance

$A \leftarrow B_1 \wedge \ldots \wedge B_n$

of a split clause and $I \models B_1 \wedge \ldots \wedge B_n$

iff I is the least fixed point of the mapping [VK76] in D'

iff I is the minimal model of D', and hence a possible model of D. □

Now, we give a rule, the *generalized closed world assumption for the possible model semantics (GCWA$_P$)*, for inferring negative information from a database.

Definition Let D be a database and F be a conjunction of ground atoms. Then $\sim F$ is inferred from D by the $GCWA_P$, if F is false in any possible model. That is, $GCWA_P(D) = \{\sim F \mid \forall M \in FP_D, M \not\models F\}$. □

2.3 Properties of Possible Model Semantics

This section discusses the relationship between the possible model semantics and previous approaches. For a definite Horn database, the following lemmas immediately follow from the definition.

Lemma 2.3.1 Let D be a definite Horn database and T_h be the mapping given in [VK76]. Then, $FP_D = \{lfp(T_h)\}$. $\Box$

Lemma 2.3.2 Let D be a definite Horn database and A be a ground atom. Then $CWA(D) \models \sim A$ iff $GCWA_P(D) \models \sim A$. $\Box$

Next, we consider an indefinite database.

Lemma 2.3.3 Let D be a database. Then $glb(FP_D)$ is a minimal model of D.

Proof FP_D makes a complete lattice, and the result follows immediately. $\Box$

Negative information inferred from the EGCWA and the GCWA$_P$ does not always coincide with each other.

Example Let $D = \{P \vee Q\}$. Then $EGCWA(D) \models \sim (P \wedge Q)$ while $GCWA_P(D) \not\models \sim (P \wedge Q)$. $\Box$

For the EGCWA, the following lemma holds.

Lemma 2.3.4 Let D be a database and F be a conjunction of ground atoms. Then $EGCWA(D) \models \sim F$ iff $\forall M \in \{glb(FP_D)\}, M \not\models F$.

Proof The result follows immediately from the definition of the EGCWA and lemma 2.3.3. $\Box$

The above lemma is also true for the GCWA, when F is a ground atom. Next, we consider the relation to the DDR.

Lemma 2.3.5 Let D be a database which contains no negative clause. Let T_d be the mapping given in [RT88], then $lfp(T_d) = lub(FP_D)$.

Proof By the definition of $lfp(T_d)$, it obviously coincides with the least upper bound of a set of possible models. $\Box$

The above lemma does not hold in the presence of negative clauses.

Example Let $D = \{P \vee Q, \ \leftarrow P \wedge Q\}$. Then $FP_D = \{\{P\}, \{Q\}\}$ whereas $lfp(T_d) = \{P, Q\}$ which is not a model of D (cf. example 4.3 in [RT88]). $\Box$

The following lemma follows from lemma 2.3.5.

Lemma 2.3.6 Let D be a database which contains no negative clause and A be a ground atom. Then

$DDR(D) \models \sim A$ iff $GCWA_P(D) \models \sim A$ $\Box$

Example Let $D = \{P \vee Q,\ \ P,\ \ \leftarrow P \wedge Q\}$. Then $DDR(D) \not\models \sim Q$ while $GCWA_P(D) \models \sim Q$. □

[LMR88] showed that the *weakly GCWA (WGCWA)* rule is also equivalent to the DDR, so we can replace the DDR by the WGCWA in the above lemma.

3 Stratified Disjunctive Databases

When there is no unique minimal model of a database, some minimal models which reflect the intended meaning of a database are often selected.

In *stratified databases*, one model is *preferred* to others by the *predicate hierarchy* in a database. There are two ways to introduce predicate hierarchy in stratified databases. [ABW88,Pr88,VG88] present such a hierarchy by allowing negative literals in the body of a clause and giving higher priorities to minimize their extension, while [RT88] gives a level mapping to predicates in a clause instead of allowing negation in the body.

Comparing these two approaches, the first seems to be more natural, as the following example shows.

Example Let $D = \{\ eligible(X) \vee disqualified(X) \leftarrow student(X),\ \ disqualified(X) \leftarrow misbehaved(X),\ \ student(John),\ \ misbehaved(John)\}$ and L be the level mapping such that $L(misbehaved) = 0$, $L(disqualified) = 1$, $L(student) = 2$ and $L(eligible) = 3$. Then we expect '$\sim eligible(John)$', but it is not deduced from D by the DDR, whereas if we present the first rule by '$eligible(X) \leftarrow student(X) \wedge \sim disqualified(X)$' then '$\sim eligible(John)$' is inferred by the ICWA [GPP86b]. □

In this section, we consider a database which contains some clauses having negative literals in their bodies.

Definition A *database* is a finite *consistent* set of the clauses of the form:

$A_1 \vee ... \vee A_m \leftarrow B_1 \wedge ... \wedge B_n$

where $m, n \geq 0$ and each A_i is an atom and each B_j is a *literal* and all variables are universally quantified at the front of the clause. □

The following definition is from [ABW88].

Definition A database D is *stratified* if there is a partition

$D = D_1 \cup ... \cup D_n$

such that the following two conditions hold for $i = 1, .., n$.

1. If a relation symbol occurs positively in a clause in D_i, then its definition (i.e, clauses which contain the relation symbol in their heads) is contained within $\bigcup_{j \leq i} D_j$.
2. If a relation symbol occurs negatively in a clause in D_i, then its definition is contained within $\bigcup_{j<i} D_j$.

D_1 can be empty. □

Note that [ABW88] gives the definition for a database which contains neither disjunctive clauses nor negative clauses, and [Pr88] permits disjunctive clauses in *locally stratified databases* but they contain no negative clause. However, the definition of stratified databases is extended to those databases containing negative clauses without any modification.

Example Let $D = \{A \vee B \leftarrow C \wedge \sim D,\ C,\ E,\ \leftarrow E \wedge \sim A\}$. Then D is stratified by $D = \{\leftarrow E \wedge \sim A\} \cup \{A \vee B \leftarrow C \wedge \sim D,\ C,\ E\}$. □

To avoid confusion in the following discussion, we use the term 'stratified disjunctive database' for a stratified database including disjunctive clauses, and otherwise we only say 'stratified database'.

Clearly, the split databases of a stratified disjunctive database become stratified databases. Then we define possible model for stratified disjunctive databases, which is an extension of the previous section.

Definition Let D be a stratified disjunctive database and M be a model of D. Then M is called *possible* if it is a *perfect model* [Pr88] of some split database of D. □

In the previous section, we defined the mapping T_D for a database containing no negative literals. However, it can easily be verified that the definition is also applicable to the database considering in this section without any modification. That is, for negative literals, $I \models \sim B$ if $B \notin I$. Accordingly, the continuous mapping $\mathbf{T_D}$ is defined in the same way and it is easily shown that the corresponding lemmas still hold.

Now we present an *iterated fixpoint semantics* for stratified disjunctive databases.

Definition Let D be a database stratified by $D_1 \cup ... \cup D_n$. Then the *iterated fixpoint semantics* of D, $ITER_D$, is defined as follows.

$\mathbf{M_1} = \{I_1 \mid I_1 \in \mathbf{T_{D_1}} \uparrow \omega(\{\emptyset\}) \text{ and } I_1 \in \mathbf{T_{D_1}}(\{I_1\})\}$,

$\mathbf{M_2} = \{I_2 \mid I_2 \in \mathbf{T_{D_2}} \uparrow \omega(\mathbf{M_1}) \text{ and } I_2 \in \mathbf{T_{D_2}}(\{I_2\})\}$,

. . .

$\mathbf{M_n} = \{I_n \mid I_n \in \mathbf{T_{D_n}} \uparrow \omega(\mathbf{M_{n-1}}) \text{ and } I_n \in \mathbf{T_{D_n}}(\{I_n\})\}$,

$ITER_D = \mathbf{M_n}$

where $\mathbf{T_{D_i}}$ is the mapping defined for D_i such that $\mathbf{T_{D_i}}(\mathbf{I}) = \bigcup_{I\in \mathbf{I}} T_{D_i}(I)$ and T_{D_i} is the mapping defined for D_i as in the previous section. □

The following theorem shows the relationship between the iterated fixpoint and the possible model semantics for stratified disjunctive databases.

Theorem 3.1.1 Let D be a stratified disjunctive database. Then I is in $ITER_D$ iff I is a possible model of D.

Proof Similar to the proof of theorem 2.2.6. □

Lemma 3.1.2 Let D be a stratified disjunctive database. Then $glb(ITER_D)$ is a perfect model of D.

Proof Perfect models are minimal elements in the iterated fixpoint, so the result follows immediately. □

For inferring negation, the GCWA$_P$ is defined in the same way as before.

Definition Let D be a stratified disjunctive database and F be a conjunction of ground atoms. Then $\sim F$ is inferred from D by the $GCWA_P$ if F is false in any possible model. That is, $GCWA_P(D) = \{\sim F \mid \forall M \subset ITER_D, M \not\models F\}$. □

Lemma 3.1.3 Let D be a stratified disjunctive database and F be a conjunction of ground atoms. Then

$ICWA(D) \models \sim F$ iff $\forall M \in \{glb(ITER_D)\}$, $M \not\models F$. □

The GCWA$_P$ and the ICWA are different in general.

Example Suppose $D = \{P \vee Q \leftarrow \sim R\}$. Then $ICWA(D) \models \sim (P \wedge Q)$, while $GCWA_P(D) \not\models \sim (P \wedge Q)$ □

4 Summary and Future Work

This paper presented a declarative semantics for disjunctive databases containing inclusive and exclusive disjunctions and introduced a new fixpoint semantics of databases in the presence of negative clauses. A disjunctive database is a database which contains some incomplete information and we consider possible models provide a more flexible way than minimal model semantics for such databases.

For inferring negative information, we introduced the GCWA$_P$ which provides an intermediate position between the GCWA and the DDR, that is, stronger than the DDR but weaker than the GCWA. In [RT88], it is proposed some criteria for the rule of inferring negative information from disjunctive databases. Applying it to the GCWA$_P$, it is *concise* and *consistent* but not necessarily *inclusive* nor *decreasing*. We consider that some of these criteria are not essential to our framework. Since a disjunction is interpreted exclusively in the presence of a negative clause, we may infer additional negative information from a database after adding new information to a database, that is, not decreasing.

Much recent work has been concerned with the semantics of *general databases* which are the extension of stratified databases [GL88,PP88,VRS88], and the extension of possible model semantics to *general disjunctive databases* is now under investigation. This paper also has not presented the procedural semantics of possible model semantics and the GCWA$_P$. It is also related to the problem of checking the satisfiability of integrity constraints in a disjunctive database. These problems, together with an efficient implementation issue, are left to future research.

Acknowledgment

We wish to thank Kazuhiro Fuchi, director of ICOT, for providing us with the opportunity to pursue this research and Shunichi Uchida, chief of the laboratory, for his encouragement.

References

[ABW88] Apt, K. R., Blair, H. A. and Walker, A., Towards a Theory of Declarative Knowledge, in *Foundations of Deductive Databases and Logic Programming* (J. Minker ed.), Morgan Kaufmann Publishers, 89-148, 1988.

[GL88] Gelfond, M. and Lifschitz, V., The Stable Model Semantics for Logic Programming, *Proc. ICLP'88*, 1070-1080, 1988.

[GPP86a] Gelfond, M., Przymusinska, H. and Przymusinski, T. C., The Extended Closed World Assumption and its Relationship to Parallel Circumscription, *Proc. PODS'86*, 133-139, 1986.

[GPP86b] Gelfond, M., Przymusinska, H. and Przymusinski, T. C., On the Relationship between Circumscription and Negation as Failure, manuscript, Dept. of Mathematical Sciences, Univ. of Texas, Univ. of Texas, 1986.

[Ll87] Lloyd, J. W., Foundations of Logic Programming, 2nd ed., Springer-Verlag, Berlin, 1987.

[LMR88] Lobo, J., Minker, J. and Rajasekar, A., Weak Completion Theory for Non-Horn Program, *Proc. ICLP'88*, 828-842, 1988.

[Lo78] Loveland, D. W., *Automated Theorem Proving: A Logical Basis*, North Holland, New York, 1978.

[Mc86] McCarthy, J., Applications of Circumscription to Formalizing Commonsence Knowledge, *Artificial Intelligence* 28, 89-118, 1986.

[Mi82] Minker, J., On Indefinite Data Bases and The Closed World Assumption, *Proc. 6th Conf. Automated Deduction, Lecture Notes in Computer Science 138*, Springer-Verlag, 292-308, 1982.

[PP88] Przymusinska, H. and Przymusinski, T. C., Weakly Perfect Model Semantics for Logic Programs, *Proc. ICLP'88*, 1106-1120, 1988.

[Pr88] Przymusinski, T. C., On the Declarative Semantics of Deductive Databases and Logic Programs, in *Foundations of Deductive Databases and Logic Programming* (J. Minker ed.), Morgan Kaufmann Publishers, 193-216, 1988.

[Re78] Reiter, R., On Closed World Databases, in *Logic and Data Bases* (H. Gallaire and J. Minker eds.), Plenum, New York, 55-76, 1978.

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

[SM83] Sakai, K. and Miyachi, T., Incorporating Naive Negation into Prolog, ICOT Technical Report, TR-28, 1983.

[SS89] Seki,H. and Sakama, C., Yet Another Semantics for Disjunctive Databases, ICOT Technical Memorandum (in preparation), 1989.

[VK76] Van Emden, M. H. and Kowalski, R. A., The Semantics of Predicate Logic as a Programming Language, *J.ACM* 23, 4, 733-742, 1976.

[VG88] Van Gelder, A., Negation as Failure using Tight Derivations for General Logic Programs, in *Foundations of Deductive Databases and Logic Programming* (J. Minker ed.), Morgan Kaufmann Publishers, 149-176, 1988.

[VRS88] Van Gelder, A., Ross, K. and Schlipf, J. S., Unfounded Sets and Well-Founded Semantics for General Logic Programs, *Proc. PODS'88* 221-230, 1988.

[YH85] Yahya, A. and Henschen, L. J., Deduction in Non-Horn Databases, *J. Automated Reasoning* 1, 141-160, 1985.