INFORMATION AND COMPUTATION **91**, 15–54 (1991)

General Logical Databases and Programs: Default Logic Semantics and Stratification

NICOLE BIDOIT* AND CHRISTINE FROIDEVAUX†

L.R.I., U.A. 410 du CNRS, Université Paris Sud, 91405 Orsay Cedex, France

Default logic is introduced as a well-suited formalism for defining the declarative semantics of deductive databases and logic programs. After presenting, in general, how to use default logic in order to define the meaning of logical databases and logic programs, the class of stratifiable databases and programs is extensively studied in this framework. Finally, the default logic approach to the declarative semantics of logical databases and programs is compared with the other major approaches. This comparison leads to showing some advantages of the default logic approach. © 1991 Academic Press, Inc.

1. INTRODUCTION

In the past few years, a number of non-monotonic logics have been introduced for default reasoning (McCarthy, 1980; McDermott and Doyle, 1980; Reiter, 1980). These logics have recently received a great deal of attention, especially in the fields of logic programming and deductive databases.

The purpose of this paper is to show that Reiter's (1980) default logic is a well-suited formalism for defining the meaning of logical databases and logic programs in a declarative fashion. Default logic provides a natural formalization of the "closed world assumption." It also provides a natural interpretation of negation, that we call negation by default. Intuitively, negation by default can be viewed as the declarative analog of negation by failure (Clark, 1978) which is essentially a computational notion. Thus, the declarative semantics of logical databases and logic programs[1] is defined here by means of the extensions of the default theories associated with these databases or programs. (Extensions of the default theory specifying a logical database are maximally consistent which is not the case for exten-

* Work performed by this author was partially supported by the PRC BD3 national project.

† Work performed by this author was partially supported by the PRC IA national project.

[1] By convention disjunction is allowed in a database but not in a logic program.

643/91/1-2

0890-5401/91 $3.00

sion of default theories in general.) However, the default theory specifying a logical database may not have an extension. In other words, a logical database may be inconsistent with respect to default logic. The class of stratifiable databases is investigated as a subclass of consistent logical databases, in the default logic framework. The concept of stratification primarily introduced in Chandra and Harel (1985) and further studied in (Apt, Blair, and Walker, 1988; Lifschitz, 1988; Naqvi, 1986; Przymusinski, 1989; Przymusinska and Przymusinski, 1988; Van Gelder, 1988) forbids recursion involving negation.

The second purpose of the paper is to compare the expressiveness of default logic with the expressiveness of other formalisms proposed in order to define the declarative semantics of logical databases and logic programs. Mainly, we lay stress on the comparison between the default logic approach and the perfect model approach (Przymusinski, 1989). This comparative study leads to the following conclusions:

(1) In general, the two approaches do not behave in the same way. In particular, the perfect model approach fails to capture the intended meaning of simple and useful programs while the default logic semantics behave satisfactory for these programs.

(2) For stratifiable databases, the two approaches are equivalent. Stratifiable databases have a quite simple structure. Indeed, roughly speaking, a stratifiable database (with negation) can be organized as a sequence of positive databases (databases without negation). This explains why the declarative semantics of a stratifiable database can be defined exclusively in terms of minimal models of subsets of this database. This also implies that concepts such as extension of default theories, supported model, preference relation (among minimal models) are unnecessarily introduced on top of the notion of minimal model while dealing with stratifiable logical databases. The equivalence between the default logic and the perfect model approaches is established through the property explained above and satisfied (only) by stratifiable databases.

For stratifiable logic programs, the equivalence of default logic, iterative fixed point approach, and pointwise circumscription is shown through the equivalence of default logic and perfect model and through the equivalence of perfect model, iterative fixed point, and pointwise circumscription established in (Przymusinski, 1989; Lifschitz, 1988).

The autoepistemic approach (Gelfond, 1987) and the stable model approach (Gelfond and Lifschitz, 1988) are also discussed versus the default logic semantics.

The paper is organized as follows. Section 2 provides a review of basic notions on default logic. In Section 3, the declarative semantics of logical databases and logic programs is investigated in the default logic framework. In Section 4, we proceed to a comparative study of the default logic approach and the other approaches proposed in the literature.

2. Default Logic

In the following, we assume that the reader is familiar with propositional and first-order logic. In order to make the discussion clear, we review some concepts of logic and introduce default logic (Reiter, 1980). We also present the notations used throughout the paper.

Let L be a first-order language. A set of formulae of L is said to be *consistent* if it has at least one (first-order) model, and is said to be *complete* if it is maximally consistent. First-order inference is denoted $\vdash$. The logical closure of a set E of formulae, denoted $Th(E)$, is $\{\alpha \mid \alpha \in L \text{ and } E \vdash \alpha\}$. By convention, a Herbrand model M of a set of formulae is represented by enumerating the ground atomic formulae true for M. Finally, let M and M' be two models of a set W of formulae, we say that M' is *less than* M, denoted $M' \leqslant M$, iff $M' \subseteq M$, and that M is a *minimal model* (Bossu and Siegel, 1985) of W iff there exists no model M' of W such that $M \neq M'$ and $M' \leqslant M$.

Default logic has been introduced by Reiter (1980) in order to formalize default reasoning. Default reasoning is a fundamental component of common sense reasoning and is a form of non-monotonic reasoning. Default logic allows one to reason about real world situations and more specifically about incomplete descriptions of real world situations using non-monotonic inferences of the form: "in absence of any information to the contrary, assume...," or "if certain information cannot be deduced from the given knowledge base, then conclude... ." In order to formally represent such a type of inference, default rules are introduced.

A *closed default rule* d over L is any expression of the form $\alpha : M\beta_1, ..., M\beta_n/\gamma$, where $n \geqslant 1$ and α, called prerequisite, $\beta_1, ..., \beta_n$, called justifications and γ, called consequent, are well-formed *closed* formulae in L. Note here that M is simply a new symbol to be read as "it is consistent." We denote by prer(d) the formula α, just(d) the set $\{\beta_1, ..., \beta_n\}$, cons($d$) the formula γ, and for a set D of default rules $\text{JUST}(D) = \bigcup_{d \in D} \text{just}(d)$. A default rule d is *normal* iff $\text{just}(d) = \{\text{cons}(d)\}$.

A (*closed*) *default theory* Δ over L consists of a pair (D, W), where W is a set of closed formulae in L (W is a possibly incompletely specified knowledge base) and D is a set of (closed) default rules over L (D is a set of meta-rules used to enrich W). In fact a default theory Δ induces a set of first-order theories, called extensions of Δ. For the purpose of our discussion, to the least fixed point definition of Reiter we prefer the more intuitive characterization:

A set E of sentences of L is an *extension* for the default theory $\Delta = (D, W)$ iff $E = \bigcup_{i=0 \cdots \infty} E_i$, where:

$E_0 = W$ and, for $i \geqslant 0$,

$E_{i+1} = Th(E_i) \cup \{\gamma \mid d \in D,\ \text{cons}(d) = \gamma,\ \text{prer}(d) \in E_i,\ \text{and}\ \forall \beta \in \text{just}(d),\ \neg\beta \notin E\}$.

Let us point out that E occurs in the definition of E_{i+1}.

The generating defaults of E with respect to Δ are: $GD(\Delta, E) = \{d \in D \mid \text{prer}(d) \in E \text{ and } \forall \beta \in \text{just}(d), \neg\beta \notin E\}$. In general, a default theory may not have extensions and if it does, these extensions may not be complete.

THEOREM 2.1 (Reiter, 1980). *Every normal default theory has an extension.*

A *default model* for Δ is a Herbrand model of an extension E for Δ. Because an extension E for Δ may not be complete in general, E may have more than one Herbrand model. This implies that a default theory Δ may have a unique extension but more than one default model.

Notation. Let $T_1 \cdots T_t$ be a sequence of sets. By convention, $T_{\rightarrow i}$ $(1 \leqslant i \leqslant t)$ denotes $\bigcup_{j=1 \cdots i} T_j$.

3. POSITIVIST DEFAULT THEORIES AND STRATIFICATION

The purpose of this section is to describe the declarative semantics of logic programs and deductive databases using Reiter's default logic. The two main issues raised by this are: for logic programs, it is largely recognized that their expressive power should be increased by the ability to express negation (Clark, 1978; Chandra and Harel, 1985; Clark and Tarnlund, 1982; Lloyd, 1987); for deductive databases, both expressing negation in the premises of rules and allowing the inference of disjunctive information have to be combined (Bidoit and Hull, 1986, 1989).

In order to make the discussion clear, the concepts and results of the paper are in general stated in the propositional case. At the end of this section, a short discussion explains how to generalize these concepts and results to encompass full first-order languages. In the following, Prop denotes a set of propositions and L the propositional language over Prop.

We first show that the "closed world assumption" is an instance of default reasoning. Roughly speaking, making the closed world assumption (CWA) corresponds to making the choice to give a description of real world situations by restricting it to positive (true) information. The CWA tremendously simplifies the representation of information. Under the CWA, the description of the information contained in the database is incomplete: it does not include negative information. However, the CWA presumes a complete knowledge about the real world situation being modeled; under the CWA, in order to derive negative information, for example, in order to derive the negative fact $\neg A$, one resorts to the following (default) inference process: "if $\neg A$ is consistent with the knowledge base, then infer that $\neg A$ is part of the knowledge base," or in terms of default rules, $:M \neg A/\neg A$.

Formally, we have

DEFINITION 3.1. Let $A \in \text{Prop}$. The *closed world default rule* associated with the proposition A is the expression $:M \neg A/\neg A$. The set of closed world default rules associated with the propositional language L, denoted CWA_L, is

$$\text{CWA}_L = \{:M \neg A/\neg A \mid A \in \text{Prop}\}.$$

CWA default reasoning is a powerful form of default reasoning. The following property illustrates that the CWA presumes full knowledge of the domain modeled: if a fact A is not known (to be true or to be false), from the absence of A in the database, we would deduce $\neg A$ which contradicts our lack of knowledge about the fact A.

LEMMA 3.1. *Let L be a propositional language and $\Delta = (D, W)$ be a default theory over L such that $\text{CWA}_L \subseteq D$ and W is consistent. Then each extension E of $\Delta = (D, W)$ is complete.*

Proof. Let E be an extension of (D, W). W consistent implies that E is consistent (Reiter, 1980). Assume that neither A nor $\neg A$ belongs to E. Because $E = Th(E)$, $E \cup \{\neg A\}$ is consistent. Hence the CWA-default rule $:M \neg A/\neg A$ can be "applied" to infer that $\neg A$ is in E, a contradiction. ∎

Because an extension E of a default theory $\Delta = (D, W)$ whose set of default rules D includes the CWA-rules is complete, E has a unique model. This implies that for a default theory $\Delta = (D, W)$ whose set of default rules D includes the CWA-rules, the set of extensions of Δ can be identified with the set of default models for Δ.

A strong correspondence between CWA-default logic and minimalism was exhibited in Bidoit and Hull (1986, 1989). For the purpose of our further discussion, we formally recall this result here:

THEOREM 3.2 (Bidoit and Hull, 1986). *Let $\Delta = (\text{CWA}_L, W)$ be a default theory over L such that W is a set of definite clauses (where the* *disjunctive form of a definite clause has at least one positive literal). The three assertions are equivalent*:

a. *E is an extension of Δ*,

b. *$E = Th(W \cup \{\neg P \mid P \in \text{Prop}$ and $\neg P \in E\})$, and*

c. *The set M of positive literals in E is a minimal (Herbrand) model for W.* ∎

Intuitively, part b says that W and the negative "facts" in E are sufficient to imply all (true) facts in E, hence E contains as much "false" facts as possible (part c). Part c of the theorem can be also read as "a default model for Δ is a minimal model for W."

We are now interested in expressing negation in the premises of deductive rules in a logical database. Assume, for example, that we want to express the following information: "If today is a week day and if John is not sick then John is at work." It is quite obvious that negation here cannot be interpreted by first-order logic negation. The formulae ($\text{WEEK\_DAY} \wedge \neg\text{SICK\_JOHN} \rightarrow \text{WORK\_JOHN}$) and ($\text{WEEK\_DAY} \rightarrow \text{SICK\_JOHN} \vee \text{WORK\_JOHN}$) are semantically equivalent in first-order logic. From the point of view of logic programming and deductive databases, it is largely recognized that this equivalence is undesirable. In fact, making the CWA leads to interpret negation not in the classical manner but as negation by default (Gabbay and Sergot, 1986). In a coherent fashion, we claim that negation in premises of deductive rules should also be interpreted as negation by default. It is clear that the meaning of "If today is a week day and if John is not sick then John is at work" is more accurately captured (under CWA) by: "if it is known that today is a week day and if there is no contradiction to assume that John is not sick then infer that John is at work." Thus, using the default logic of Reiter, we would write: $\text{WEEK\_DAY} : M\neg\text{SICK\_JOHN}/\text{WORK\_JOHN}$.

We would like to point out here that negation by default and negation as failure (Clark, 1978) are slightly different. As explained in Gabbay and Sergot (1986), the notion of negation as failure is a computational notion. It is based on a resolution procedure and on the concept of failure of this procedure. On the other hand, the notion of negation by default is a logical one. It is based on the classical notion of consistency.

The general form of default rules used to express deductive rules with negative premises is defined below:

Definition 3.2. A default rule d over L is a *positivist default rule* if prer(d) is a conjunction (possibly empty) of positive literals, just(d) is a nonempty set of negative literals, and cons(d) is a disjunction of positive literals (at least one).

We now are able to syntactically specify logical databases and logic programs in the framework of default logic. Logical databases and logic programs are mapped on a specific class of default theories called positivist default theories whose definition follows:

DEFINITION 3.3. A default theory $\Delta = (D, W)$ over a language L is a *positivist default theory* iff $\mathrm{CWA}_L \subseteq D$, $D - \mathrm{CWA}_L$ is a set of positivist default rules, and W is a set of definite clauses.

CONVENTION. In the following, by convention:

"logical database" designates a positivist default theory, in general and

"logic program" designates more specifically a positivist default theory $\Delta = (D, W)$ without disjunction; i.e., consequence of any positivist default rule in D is reduced to a single literal and W is a set of definite Horn clauses.

PRINCIPLE. In the following, the declarative semantics of a logical database/logic program is defined by means of the set of extensions of the positivist default theory specifying the database/program or in an equivalent way by means of the set of default models of the positivist default theory specifying the database/program.

Recall here that the set of default rules of a positivist default theory Δ includes the CWA-rules and thus the set of extensions of Δ can be identified with the set of default models for Δ.

EXAMPLE 3.1 (A logical database). Assume we want to express the following information about someone named John:

Today is a weak day.

If today is a weak day and if John is not sick then John is at work.

If John is at work then he is teaching or attending a meeting.

This database is specified by the following positivist default theory (D, W) below.

$:M\ \neg$ WEEK_DAY/$\neg$WEEK_DAY $:M\ \neg$ SICK_JOHN/$\neg$SICK_JOHN $:M\ \neg$ WORK_JOHN/$\neg$WORK_JOHN	CWA-default rules
WEEK_DAY : $M\ \neg$ SICK_JOHN/WORK_JOHN	WEEK_DAY WORK_JOHN $\rightarrow$ TEACH_JOHN $\vee$ meeting_JOHN
Positivist default rules D	Positive clauses W

It is easy to verify that the positivist default theory (D, W) admits two extensions:

$$E = Th(W \cup \{\text{WEEK\_DAY}, \neg\text{SICK\_JOHN}, \text{WORK\_JOHN}, \text{TEACH\_JOHN}, \neg\text{MEETING\_JOHN}\}),$$

$$E' = Th(W \cup \{\text{WEEK\_DAY}, \neg\text{SICK\_JOHN}, \text{WORK\_JOHN}, \neg\text{TEACH\_JOHN}, \text{MEETING\_JOHN}\}).$$

Using default logic allows us to clearly distinguish rules that involve negation from those that do not. This distinction is of particular interest because first-order logic is not sufficient for defining the semantics of the first kind of rules, while it is sufficient for the second kind.

Let us discuss two more examples in order to illustrate well-known problems that arise when one increases the expressive power of logical databases and logic programs with negation.

EXAMPLE 3.2. Consider the two positivist default theories $\Delta_1 = (D_1, W_1)$ and $\Delta_2 = (D_2, W_2)$ where:

$$D_1 = \{:M \neg A/\neg A,\ :M \neg B/\neg B,\ :M \neg A/B,\ :M \neg B/A\}, \quad W_1 = \{\ \}; \text{ and}$$

$$D_2 = \{:M \neg A/\neg A,\ :M \neg B/\neg B,\ :M \neg A/A\}, \quad W_2 = \{B\}.$$

Δ_1 has two extensions, namely $Th(\{\neg A, B\})$ and $Th(\{A, \neg B\})$ but Δ_2 has no extension.

The example above shows that a positivist default theory (Δ_2) may not have an extension: "*inconsistent*" programs can be written. Intuitively, this arises when a certain type of recursion occurs on negation. Note that recursion involving negation does not necessarily lead to inconsistencies in our framework. The default theory Δ_1 does not contain (syntactically) any disjunctive consequences, neither in D_1 nor in W_1. Intuitively, this suggests that Δ_1 does not contain incomplete informations. However, Δ_1 has two extensions (as a matter of fact the same extensions as the "disjunctive" default theory $(\{:M \neg A/\neg A,\ :M \neg B/\neg B\}, \{A \vee B\})$. In this case, we say that the logic program Δ_1 is "*ambiguous*."

At this point of the presentation, it is interesting to examine the various semantics associated by other approaches with the logic program Δ_2 of Example 3.2. Recall that under default logic interpretation, this program is considered to be inconsistent.

(a) Following the perfect model approach (Przymusinski, 1988) further discussed in Section 4, this logic program would be specified by the set $\{B, \neg A \rightarrow A\}$ of P-clauses and has a unique perfect model $\{A, B\}$. Note that following this approach, negation is surprisingly "interpreted" in this case by first-order negation and that this program is equivalent to $\{A \vee A, B\} = \{A, B\}$.

(b) Following the autoepistemic approach (Gelfond, 1987; Moore, 1985) further discussed in Section 4, this program would be specified by the autoepistemic theory $\{B, \neg LA \rightarrow A\}$ and does not have stable expansions.

(c) Following the default logic approach of Lukaszewicz (1988) which is a variant of Reiter's default logic, the default theory used to specify this program is still Δ_2 but in this case, Δ_2 has a unique m-extension (modified extension) $Th(\{B, \neg A\})$ with respect to $F = \{\neg A\}$.

(d) Finally, following the completion approach (Clark, 1978), the completion of this program is $\{B \leftrightarrow \text{true}, \neg A \leftrightarrow A\}$. It is inconsistent.

In the following, we restrict our attention to consistent and non-ambiguous logical databases/logic programs, that is, to positivist default theories having at least one extension or equivalently having at least one default model. More precisely, among these consistent and non-ambiguous logical databases, we are going to investigate the class of stratifiable logical databases. Stratification has been first introduced in Chandra and Harel (1985), then investigated in Apt *et al.* (1988), Naqvi (1986), and Van Gelder (1988) and also extended by Przymusinski (1988). Roughly speaking, stratification forbids recursion involving negation.

DEFINITION 3.4. A positivist default theory $\Delta = (D, W)$ over the language L is *stratifiable* iff there exists a sequence $S = \text{Prop}_1 \cdots \text{Prop}_s$, called a *stratification* for Δ, such that $\text{Prop}_1 \cdots \text{Prop}_s$ is a partition of the set Prop associated with L and satisfying:

(i) For each $(P_1 \wedge \cdots \wedge P_p : M \neg Q_1, ..., M \neg Q_q / R_1 \vee \cdots \vee R_r)$ in D, there exists k in $[1 \cdots s]$ such that:

$$\forall j \in [1 \cdots r],\ R_j \in \text{Prop}_k,$$
$$\forall j \in [1 \cdots p],\ \exists 1 \leqslant i \leqslant k \mid P_j \in \text{Prop}_i,$$
$$\forall j \in [1 \cdots q],\ \exists 1 \leqslant i < k \mid Q_j \in \text{Prop}_i;$$

and

(ii) for each $(P_1 \wedge \cdots \wedge P_p \rightarrow R_1 \vee \cdots \vee R_r) \in W$, there exists k in $[1 \cdots s]$ such that:

$$\forall j \in [1 \cdots r],\ R_j \in \text{Prop}_k,$$
$$\forall j \in [1 \cdots p],\ \exists 1 \leqslant i \leqslant k \mid P_j \in \text{Prop}_i.$$

EXAMPLE 3.1 (Continued). Two possible stratifications for (D, W) are given below:

$S = \mathrm{Prop}_1 \mathrm{Prop}_2$ is a stratification for (D, W) with:

$\mathrm{Prop}_1 = \{\textsc{sick\_john}\}$,

$\mathrm{Prop}_2 = \{\textsc{week\_day}, \textsc{work\_john}, \textsc{teach\_john}, \textsc{meeting\_john}\}$,

and

$S' = \mathrm{Prop}'_1 \mathrm{Prop}'_2 \mathrm{Prop}'_3 \mathrm{Prop}'_4$ is also a stratification for (D, W) with:

$\mathrm{Prop}'_1 = \{\textsc{sick\_john}\}$, $\mathrm{Prop}'_2 = \{\textsc{week\_day}\}$

$\mathrm{Prop}'_3 = \{\textsc{work\_john}\}$, $\mathrm{Prop}'_4 = \{\textsc{teach\_john}, \textsc{meeting\_john}\}$.

This example shows that there may be several different stratifications for a positivist default theory. Note also that in our example, the "smallest" (in the number of strata) stratification for (D, W) has two strata. Intuitively, (D, W) is stratifiable because the "definition" of WORK_JOHN in which SICK_JOHN occurs negatively can be totally "isolated" from the "definition" of SICK_JOHN. The notion of stratification implies a hierarchy of the "definitions" of the propositions.

We just mention here without development that the problem of deciding whether a database is stratifiable, is polynomial. The algorithm proposed in Apt *et al.* (1988) takes into account logic programs (no disjunction) but is easily generalizable to check whether a logical database (with disjunction) is stratifiable.

Notation. Given a stratifiable positivist default theory (D, W) and a stratification $S = \mathrm{Prop}_1 \cdots \mathrm{Prop}_s$ for (D, W), S induces a partitioning on D and W. In the following, when S is understood, we denote by L_i the propositional language over the set Prop_i of propositions. Also, when S is understood, we denote by $D_1 \cdots D_s$ (resp., $W_1 \cdots W_s$) the elements of the partition induced by S on D (resp., on W). For $i = 1 \cdots s$, $D_i = \{d \mid d \in D$ and $\mathrm{cons}(d) \in L_i\}$ and $W_i = \{\alpha \to \beta \mid \alpha \to \beta \in W$ and $\beta \in L_i\}$. In the following, (D_i, W_i) is called the ith *stratum* induced by S on (D, W). When S is understood, CWA_i denotes the set of closed world default rules associated with the sublanguage L_i of L. Note in particular, that for any stratification S for (D, W), $D_1 = \mathrm{CWA}_1$.

Actually, an alternative definition of the meaning of a *stratifiable* logical database can be easily and naturally obtained by utilizing the hierarchy of the database induced by any stratification. Such an alternative definition is proposed and shown equivalent to the general default semantics for logical databases (Principle). Given a stratification S of a stratifiable positivist default theory Δ, intuitively we know that, the negative premises of a deductive rule in the ith stratum of Δ (that is, the justifications of default rules in D_i) are "completely defined" at some level j such that $j < i$. Thus we propose to consider successively each stratum induced by S on Δ and propagate the information obtained at the ith level while examining the $(i+1)$th stratum. This leads to an alternative definition of the semantics of logical databases and logic programs based on the structure of these databases or programs. However, we show in the further discussion that this definition is independent of a given stratification.

DEFINITION 3.5. Let Δ be a positivist default theory over the language L such that Δ is stratifiable. Let $S = \text{Prop}_1 \cdots \text{Prop}_s$ be a stratification for Δ. A set E of sentences over L is an *extension of Δ with respect to S* iff there exists a sequence $E_1, \ldots, E_s$ such that:

(i) $E = E_s$, and

(ii) $\forall i \in [1 \cdots s]$, E_i is an extension of $\Delta_i = (D_i, W_i \cup E_{i-1})$, assuming $E_0 = \varnothing$.

EXAMPLE 3.1 (Continued). Consider the already given stratification S. The first stratum ($D_1 = \{:M \neg \text{SICK\_JOHN}/\neg \text{SICK\_JOHN}\}$, $W_1 = \varnothing$) induced by S on (D, W) has a unique extension, namely:

$$E_1 = \{\neg \text{SICK\_JOHN}\},$$

Now let us propagate E_1 in the second stratum (D_2, W_2) induced by S on (D, W), that is let us consider the default theory (D_2, W'_2), where:

$$\begin{aligned} D_2 = \{&:M \neg \text{WEEK\_DAY}/\neg \text{WEEK\_DAY},\ :M \neg \text{WORK\_JOHN}/\neg \text{WORK\_JOHN},\\ &:M \neg \text{TEACH\_JOHN}/\neg \text{TEACH\_JOHN},\\ &:M \neg \text{MEETING\_JOHN}/\neg \text{MEETING\_JOHN}\\ &\quad \text{WEEK\_DAY} :M \neg \text{SICK\_JOHN}/\text{WORK\_JOHN}\} \end{aligned}$$

$$\begin{aligned} W'_2 = \{&\text{WEEK\_DAY},\ \text{WORK\_JOHN} \to \text{TEACH\_JOHN} \vee \text{MEETING\_JOHN}\}\\ &\cup \{\neg \text{SICK\_JOHN}\}. \end{aligned}$$

This default theory has two extensions, namely:

$$\begin{aligned} E_2 = Th(\{&\text{WEEK\_DAY},\ \neg \text{SICK\_JOHN},\ \text{WORK\_JOHN},\\ &\text{TEACH\_JOHN},\ \neg \text{MEETING\_JOHN}\}),\\ E'_2 = Th(\{&\text{WEEK\_DAY},\ \neg \text{SICK\_JOHN},\ \text{WORK\_JOHN},\\ &\neg \text{TEACH\_JOHN},\ \text{MEETING\_JOHN}\}). \end{aligned}$$

Thus E_2 and E'_2 are two extensions of (D, W) with respect to the stratification S. Note that they respectively coincide with the already given extensions E and E' of (D, W).

We now are going to show that a stratifiable positivist default theory has at least one extension with respect to any stratification. Then we argue that for a stratifiable positivist default theory, the notion of extension (*as defined by Reiter*) and the notion of extension *with respect to* a stratification are strictly equivalent. From this, we easily deduce that the concept of extension with respect to a stratification is independent of the given stratification and, moreover, that the stratification property is sufficient to conclude the existence of canonical extensions for a positivist default theory.

In order to proceed with this study, we need some intermediate results. Roughly speaking, the next technical lemma states that given a positivist default theory (D, W), if justifications of non-CWA-default rules in D have their thruth value totally determined by the contents of W, then these non-CWA-default rules can be turned into clauses that are added to W.

LEMMA 3.3. *Let $\Delta = (D, W)$ be a default theory over the language L such that D is a set of positivist default rules including* CWA_L *and W is a set of clauses. Assume that $\forall(\neg Q) \in \mathrm{JUST}(D - \mathrm{CWA}_L)$, $Q \in W$ or $\neg Q \in W$. Then E is an extension of Δ iff E is an extension of* $(\mathrm{CWA}_L,\ W \cup W')$, *where*:

$$W' = \{\alpha \to \gamma \mid d \in D, \alpha = \mathrm{prer}(d), \gamma = \mathrm{cons}(d), \textit{and}$$
$$\forall(\neg Q) \in \mathrm{just}(d), (\neg Q) \in W\}.$$

Sketch of Proof. ($\Rightarrow$) Assume that E is an extension of Δ. Let T denote $Th(W \cup W' \cup \{\neg Q \mid \neg Q \in E\})$. To show that E is an extension of $\Delta' = (\mathrm{CWA}_L, W \cup W')$, it suffices, by Theorem 3.2b, to show that $E = T$. By noticing that $W \subseteq E$ and by definition of W', we directly obtain that $T \subseteq E$.

In order to prove that $E = \bigcup_{i=0 \cdots \infty} E_i \subseteq T$, we prove, by induction on i, that $E_i \subseteq T$, where $E_0 = W$, and for $i \geqslant 0$, $E_{i+1} = Th(E_i) \cup F_i$ with $F_i = \{\mathrm{cons}(d) \mid d \in D, \mathrm{prer}(d) \in E_i, \forall(\neg Q) \in \mathrm{just}(d), (\neg Q) \in E\}$.

($\Leftarrow$) Assume that E' is an extension of $\Delta' = (\mathrm{CWA}_L, W \cup W')$. To prove that E' is an extension of $\Delta = (D, W)$, we show that $E' = \bigcup_{i=0 \cdots \infty} E_i$, where $E_0 = W$, and for $i \geqslant 0$, $E_{i+1} = Th(E_i) \cup F_i$ with $F_i = \{\mathrm{cons}(d) \mid d \in D, \mathrm{prer}(d) \in E_i, \forall(\neg Q) \in \mathrm{just}(d), (\neg Q) \in E'\}$.

We note that $F_0 = \{\neg Q \mid (\neg Q) \in W\} \cup \{\gamma \mid \alpha \to \gamma \in W', \alpha \in W\}$ and for $i \geqslant 1$, $F_i = \{\gamma \mid \alpha \to \gamma \in W', \alpha \in E_i\}$. On the one hand, by induction on i, we show that $E_i \subseteq E'$. On the other hand, using Theorem 3.2b, we show that $E' = Th(W \cup W' \cup \{\neg Q \mid \neg Q \in E'\}) \subseteq \bigcup_{i=1 \cdots \infty} E_i$. ∎

The next result provides a variant of the definition of an extension with respect to a stratification. This result makes use of the previous lemma.

COROLLARY 3.4. *Let Δ be a positivist default theory over the language L and assume that $S = \mathrm{Prop}_1 \cdots \mathrm{Prop}_s$ is a stratification for Δ. Then E is an extension of Δ with respect to S iff there exists a sequence $E_1, ..., E_s$ such that*:

(i) *$E = E_s$ and*

(ii) *$\forall i \in [1 \cdots s]$, E_i is an extension of* $(\mathrm{CWA}_{\to i}, V_i)$ *with*:

$$V_i = \bigcup_{j=1 \cdots i} (W_j \cup W_j')$$

and, for $j = 1 \cdots s$,

$$W_j' = \{\alpha \to \gamma \mid d \in D_j, \alpha = \mathrm{prer}(d), \gamma = \mathrm{cons}(d) \text{ and}, \\ \forall (\neg Q) \in \mathrm{just}(d), (\neg Q) \in E_{j-1}\},$$

assuming $E_0 = \varnothing$.

Proof.

CLAIM 3.1. *Let $E_1, ..., E_s$ be a sequence such that for $i = 1 \cdots s$, E_i is a set of sentences of $L_{\to i}$. Then, $E_1, ..., E_s$ satisfies condition* (ii) *of Definition* 3.5 *iff $E_1, ..., E_s$ satisfies condition* (ii) *of Corollary* 3.4.

The proof of the claim is presented in Appendix A. ▌

The proof of Corollary 3.4 is immediate from the above claim. ▌

This last result is of interest because it gives a definition of an extension with respect to a stratification in terms of extensions of CWA-default theories. Then we know that the extensions of a CWA-default theory (D, W) can be identified with the minimal models of W. In fact, Corollary 3.4 together with Theorem 3.2c leads us to expect that minimalism is enough to define the semantics of stratifiable logical databases and programs (Lemma 4.1.1 and 4.1.2 of Section 4).

We now simply invoke Corollary 3.4 to prove the two main results of this section: the first one shows that stratifiable programs or databases are consistent in the sense that, given a stratification S, an extension with respect to S can be exhibited for these programs or databases; the second one states the equivalence between Reiter's notion of extension and the notion of extension with respect to a stratification.

THEOREM 3.5. *Let Δ be a positivist default theory over the language L and assume that S is a stratification for Δ. Then,*

Δ has an extension with respect to S.

Proof. Assuming $S = \mathrm{Prop}_1 \cdots \mathrm{Prop}_s$, it suffices to show that $\forall i \in [1 \cdots s]$, $(D_i, W_i \cup E_{i-1})$—recall that E_0 is assumed to be empty—has an extension or equivalently, that $\forall i \in [1 \cdots s]$, $(\mathrm{CWA}_{\to i}, V_i)$ has an extension, where V_i is defined in Corollary 3.4. Note that $\forall i \in [1 \cdots s]$, $(\mathrm{CWA}_{\to i}, V_i)$ is a closed normal default theory, therefore by Theorem 2.1, it has an extension. ▌

Again using Corollary 3.4, we have

THEOREM 3.6. *Let Δ be a positivist default theory over the language L and assume that S is a stratification for Δ. Then*:

E is an extension of Δ iff E is an extension of Δ with respect to S.

Proof. ($\Leftarrow$) Let E be an extension of Δ with respect to $S = \mathrm{Prop}_1 \cdots \mathrm{Prop}_s$. We show that E is an extension of Δ, that is, $E = \bigcup_{i=0 \cdots \infty} G_i$, where $G_0 = W$, and $G_{i+1} = Th(G_i) \cup T_i$ with $T_i = \{\mathrm{cons}(d) \mid d \in D, \mathrm{prer}(d) \in G_i$, and $\forall (\neg Q) \in \mathrm{just}(d), \neg Q \in E\}$. By hypothesis, $E = E_s$, where for $j \in [1 \cdots s]$ E_j is an extension of $(D_j, W_j \cup E_{j-1})$ (recall that $E_0 = \varnothing$) (or equivalently, extension of $(\mathrm{CWA}_{\to j}, V_j)$, where V_j is defined in Corollary 3.4).

a. Let us show first that $\bigcup_{i=0 \cdots \infty} G_i \subseteq E_s$. In order to do this, we show that $\forall i \in [0 \cdots \infty[$, $\exists j \in [1 \cdots s]$, $G_i \subseteq E_j$. We proceed by induction on i:

i = 0: $G_0 = W$, E_s is an extension of $(\mathrm{CWA}_{\to s}, V_s)$ and $W \subseteq V_s \subseteq E_s$.

Induction step. Assume now that $\forall i \leqslant k$, $\exists j \in [1 \cdots s]$, $G_i \subseteq E_j$. Consider $G_{k+1} = Th(G_k) \cup T_k$. By induction hypothesis, let $j \in [1 \cdots s]$ be such that $G_k \subseteq E_j$. E_j extension of $(\mathrm{CWA}_{\to j}, V_j)$ is complete over $L_{\to j}$, thus $Th(G_k) \subseteq Th(E_j) = E_j$. Now let γ be in T_k (i.e.) let γ be such that $\exists d \in D$, $\mathrm{cons}(d) = \gamma$, $\mathrm{prer}(d) \in G_k$, and $\forall (\neg Q) \in \mathrm{just}(d)$, $\neg Q \in E$. Then, $\mathrm{prer}(d) \in E_j$. Let j' be such that $\forall (\neg Q) \in \mathrm{just}(d)$, $\neg Q \in E_{j'}$ and $j0 = \max(j, j')$. It is clear that $(\mathrm{prer}(d) \to \gamma) \in W'_{j0+1}$ and thus $\gamma \in Th(W'_{j0+1} \cup E_j) \subseteq E_{j0+1} \subseteq E_s$. In conclusion, $G_{k+1} \subseteq E_s$.

b. It remains to show that $E_s \subseteq \bigcup_{i=0 \cdots \infty} G_i$. In order to do this, we prove that $\forall j \in [1 \cdots s]$, $E_j \subseteq \bigcup_{i=0 \cdots \infty} G_i$. We proceed by induction on j:

j = 1: $E_1 \subseteq G_1$ is immediate.

Induction step. Suppose now that $\forall j \leqslant k$, $E_j \subseteq \bigcup_{i=0 \cdots \infty} G_i$. Let E_{k+1} be an extension of $(\mathrm{CWA}_{\to k+1}, W_{k+1} \cup W'_{k+1} \cup E_k)$. By Theorem 3.2b, $E_{k+1} = Th(W_{k+1} \cup W'_{k+1} \cup E_k \cup \{\neg Q \mid \neg Q \in E_{k+1}\})$. Using arguments similar to the ones used in the proof of Lemma 3.3 (part ($\Rightarrow$)a), we easily show that

$$W_{k+1} \cup W'_{k+1} \cup \{\neg Q \mid \neg Q \in E_{k+1}\} \subseteq \bigcup_{i=0\cdots\infty} G_i.$$

On the other hand, by induction hypothesis, we have that $E_k \subseteq \bigcup_{i=0\cdots\infty} G_i$. Thus,

$$E_{k+1} \subseteq Th\left(\bigcup_{i=0\cdots\infty} G_i\right) = \bigcup_{i=0\cdots\infty} G_i.$$

($\Rightarrow$) Assume now that E is an extension of Δ. Then $E = \bigcup_{i=0\cdots\infty} G_i$, where $G_0 = W$ and $G_{i+1} = Th(G_i) \cup T_i$ with $T_i = \{\text{cons}(d) \mid d \in D, \text{prer}(d) \in G_i, \forall(\neg Q) \in \text{just}(d), \neg Q \in E\}$. We need an intermediate result to continue the proof:

Claim 3.2. $\forall i \in [1 \cdots \infty[, \forall k \in [1 \cdots s], Th(G_i) \mid L_{\to k} = Th(G_i \mid L_{\to k})$.

The proof of the claim can be found in Appendix B.

To show that E is an extension of Δ with respect to S, we show that for $t = 1 \cdots s$, $E_t = E \mid L_{\to t}$ is an extension of $(D_t, W_t \cup E_{t-1})$, where $E_0 = \emptyset$. We proceed by induction on t:

$$\mathbf{t=1:}\ E_1 = \bigcup_{i=0\cdots\infty} (G_i \mid L_{\to 1}).$$

Let us show that $E_1 = Th(W_1 \cup \{\neg Q \mid \neg Q \in E_1\})$:

a. Clearly, $Th(W_1 \cup \{\neg Q \mid \neg Q \in E_1\}) \subseteq E_1$.

b. It remains to show that $E_1 \subseteq Th(W_1 \cup \{\neg Q \mid \neg Q \in E_1\})$(i.e.) $\forall i \in [1 \cdots \infty[$,

$$G_i \mid L_{\to 1} \subseteq Th(W_1 \cup \{\neg Q \mid \neg Q \in E_1\}).$$

$$G_0 \mid L_{\to 1} = W_1 \subseteq Th(W_1 \cup \{\neg Q \mid \neg Q \in E_1\}),$$

$$G_1 \mid L_{\to 1} = Th(G_0) \mid L_{\to 1} \cup T_1 \mid L_{\to 1}$$

with $T_1 \mid L_{\to 1} = \{\text{cons}(d) \mid d \in D_1, \text{prer}(d) \in G_0, \text{and } \forall(\neg Q) \in \text{just}(d), \neg Q \in E\}$. Because $D_1 = \text{CWA}_1$, $T_1 \mid L_{\to 1} = \{\neg Q \mid \neg Q \in E_1\}$ and from Claim 3.2, we have $Th(G_0) \mid L_{\to 1} \subseteq Th(G_0 \mid L_{\to 1}) = Th(W_1)$ and thus:

$$G_1 \mid L_{\to 1} \subseteq Th(W_1 \cup \{\neg Q \mid \neg Q \in E_1\}).$$

Finally, for $i \geqslant 2$, it is easy to see that:

$$G_i \mid L_{\to 1} = Th(G_1 \mid L_{\to 1}) \subseteq Th(W_1 \cup \{\neg Q \mid \neg Q \in E_1\}).$$

In conclusion, $E_1 \subseteq Th(W_1 \cup \{\neg Q \mid \neg Q \in E_1\})$.

Induction step. Assume that

$$\text{for } 1 \leqslant t \leqslant k,\ E_t \text{ is an extension of } (D_t, W_t \cup E_{t-1}) \text{ with } E_0 = \varnothing. \quad (\dagger)$$

Consider $E_{k+1} = E \mid L_{\to k+1}$ and let us show that $E_{k+1} = \bigcup_{j=0\cdots\infty} H_j$, where $H_0 = W_{k+1} \cup E_k$ and $H_{j+1} = Th(H_j) \cup S_j$ with

$$S_j = \{\text{cons}(d) \mid d \in D_{k+1},\ \text{prer}(d) \in H_j \text{ and } \forall(\neg Q) \in \text{just}(d),\ \neg Q \in E_{k+1}\}.$$

a. Let us first show that $\bigcup_{j=0\cdots\infty} H_j \subseteq E_{k+1}$, that is, $\forall j \in [0 \cdots \infty[$, $H_j \subseteq E_{k+1}$. We proceed by induction on j:

j = 0. $H_0 = W_{k+1} \cup E_k$ and by definition $W_{k+1} = W \mid L_{\to k+1} \subseteq E_{k+1}$. On the other hand, $E_k \subseteq E_{k+1}$ thus $H_0 \subseteq E_{k+1}$.

induction step. Assume now that for $j \leqslant h$, $H_j \subseteq E_{k+1}$. Let us consider $H_{h+1} = Th(H_h) \cup S_h$ and show that $H_{h+1} \subseteq E_{k+1}$. By induction hypothesis, $H_h \subseteq E_{k+1}$ and because E_{k+1} is complete over $L_{\to k+1}$: $Th(H_h) \subseteq Th(E_{k+1}) = E_{k+1}$. Let $\gamma \in S_h$, then there exists $d \in D_{k+1}$ such that $\text{cons}(d) = \gamma$, $\text{prer}(d) \in H_h$, and $\forall(\neg Q) \in \text{just}(d)$, $\neg Q \in E_{k+1}$. From $H_h \subseteq E_{k+1}$, we deduce that $\text{prer}(d) \in E_{k+1} \subseteq E$; and from $(\neg Q) \in E_{k+1} \subseteq E$, $\neg Q \in E$. Thus $d \in GD(\Delta, E)$ and $\gamma \in E$; $\gamma \in E$ and $d \in D_{k+1}$ implies that $\gamma \in E_{k+1}$. In conclusion, $H_{h+1} \subseteq E_{k+1}$ and $\bigcup_{j=0\cdots\infty} H_j \subseteq E_{k+1}$.

b. It remains to show that $E_{k+1} \subseteq \bigcup_{j=0\cdots\infty} H_j$ (i.e.) $\bigcup_{i=0\cdots\infty}(G_i \mid L_{\to k+1}) \subseteq \bigcup_{j=0\cdots\infty} H_j$. We show by induction on i that $\forall i \in [0 \cdots \infty[$, $G_i \mid L_{\to k+1} \subseteq \bigcup_{j=0\cdots\infty} H_j$.

i = 0. $G_0 \mid L_{\to k+1} = W_{\to k+1} = W_{\to k} \cup W_{k+1}$. By induction hypothesis ($\dagger$), $W_{\to k} \subseteq E_k \subseteq H_0$. On the other hand, $W_{k+1} \subseteq H_0$. Thus $W_{\to k+1} \subseteq H_0$.

induction step. Assume now that $\forall i \leqslant p$, $G_i \mid L_{\to k+1} \subseteq \bigcup_{j=0\cdots\infty} H_j$. Consider $\gamma \in G_{p+1} \mid L_{\to k+1}$ and let us show that $\gamma \in \bigcup_{j=0\cdots\infty} H_j$: $G_{p+1} \mid L_{\to k+1} = Th(G_p) \mid L_{\to k+1} \cup T_p \mid L_{\to k+1}$. If $\gamma \in Th(G_p) \mid L_{\to k+1}$, then by Claim 3.2, $\gamma \in Th(G_p \mid L_{\to k+1})$ and then by induction hypothesis, $\gamma \in Th(\bigcup_{j=0\cdots\infty} H_j) = \bigcup_{j=0\cdots\infty} H_j$. Now let $\gamma \in (T_p) \mid L_{\to k+1}$: $(T_p) \mid L_{\to k+1} = (\{\omega \mid d \in D,\ \omega = \text{cons}(d),\ \text{prer}(d) \in G_p, \text{ and } \forall(\neg Q) \in \text{just}(d),\ \neg Q \in E\}) \mid L_{\to k+1}$ (i.e., $(T_p) \mid L_{\to k+1} = \{\omega \mid d \in D_{\to k+1},\ \omega = \text{cons}(d),\ \text{prer}(d) \in G_p, \text{ and } \forall(\neg Q) \in \text{just}(d),\ \neg Q \in E\}$. Because Δ is stratifiable, $d \in D_{\to k+1}$ implies $\text{prer}(d) \in G_p \mid L_{\to k+1}$. Two cases arise:

Case 1. $d \in D_{k+1}$; then by induction hypothesis from $\text{prer}(d) \in G_p \mid L_{\rightarrow k+1}$ we have that: $\exists j0 \in [0 \cdots \infty[$, $G_p \mid L_{\rightarrow k+1} \subseteq H_{j0}$ and thus (a$_1$) $\exists j0 \in [0 \cdots \infty[$, $\text{prer}(d) \in H_{j0}$. On the other hand, $d \in D_{k+1}$ implies (b$_1$) $\forall (\neg Q) \in \text{just}(d)$, $\neg Q \in E \mid L_{\rightarrow k+1}$. From (a$_1$) and (b$_1$), we obtain that $\gamma = \text{cons}(d) \in H_{j0+1}$.

Case 2. $d \in D_j$, $j \leqslant k$. Then $\text{prer}(d) \in G_p \subseteq E$ and $\forall (\neg Q) \in \text{just}(d)$, $\neg Q \in E$, thus $d \in GD(\Delta, E)$, and $\gamma \in E$. Finally, $d \in D_j$ implies $\gamma \in E_j \subseteq E_k \subseteq H_0$.

In conclusion, $\gamma \in \bigcup_{j=0 \cdots \infty} H_j$. ∎

From Theorems 3.5 and 3.6, we immediately deduce

COROLLARY 3.7. *Let Δ be a positivist default theory over the language L such that Δ is stratifiable. Then: E is an extension of Δ with respect to S iff E is an extension of Δ with respect to S′ where S and S′ are two distinct stratifications for Δ.*

More important is the following:

THEOREM 3.8. *Let Δ be a positivist default theory over the language L such that Δ is stratifiable. Then: There exists a set E of formulae over L such that E is an extension of Δ (with respect to Reiter's definition).*

Once again, we emphasize the fact that the stratification constraint on the positivist default theories is a sufficient but not necessary condition for the existence of an extension; recall that the non-stratifiable default theory Δ_1 of Example 3.2 has two extensions. Stratification also is not a necessary condition for the uniqueness of the extensions: in Section 4, a non-stratifiable program is given in Example 4.1.4 which has a unique extension.

As a matter of fact, we would like to lay stress on the significance of Theorem 3.8 from the default logic point of view. Characterizing a class of default theories that have at least one extension is a problem that received a lot of attention but few and sometimes disappointing solutions. Reiter has only established the existence of extensions for normal theories (Theorem 2.1) whose semantics is given in Lukaszewicz (1985). Unfortunately, the class of normal theories has turned out to be insuffucient for practical applications like knowledge representation: semi-normal default theories whose default rules are of the form $\alpha : M(\beta \wedge \gamma)/\gamma$, have been introduced by Reiter and Criscuolo (1981). Although some semi-normal default theories (those corresponding to acyclic inheritance networks with exceptions) have an extension (Etherington and Reiter, 1983), in general they need not have extensions. In Froidevaux (1986), an other class of default theories, called taxonomic default theories, is defined whose default rules are neither normal nor semi-normal. Roughly speaking, "taxonomy" rules out recursion completely and is not concerned with the CWA. In conclusion, the class of stratifiable positivist default theories for which the existence of extensions has been showed, extends the class of taxonomic default theories. From a practical point of view, the class of stratifiable positivist default theories is of great interest.

In this section, we have defined the declarative semantics of logical databases and logic programs using default logic. For the class of stratifiable positivist default theories, we have provided an alternative definition of the concept of extension that utilizes the notion of stratification. We have shown that stratifiable logical databases and thus logic programs (positivist default theories without disjunction) are consistent. We have not yet proved that logic programs are non-ambiguous, i.e., that logic programs have exactly one extension. However, via the equivalence between the default logic approach and the perfect model approach which is discussed in the next section, and since stratifiable logic programs have a unique perfect model, non-ambiguity of stratifiable logic programs can be established.

To complete the presentation of default semantics for logical databases, we briefly discuss how the definitions and results presented up to now can be generalized in order to deal with first order languages.

Consider a finite (non-empty) set Const of constant symbols, a finite set Fun of function symbols, an infinite set Var of variable symbols and a finite set Pred of predicate symbols. In the following, L denotes the first-order language over the alphabet Const $\cup$ Fun $\cup$ Var $\cup$ Pred. We denote by H_L (or simply H when L is understood) the Herbrand base of L.

In this context, the CWA is expressed using the following set of default rules:

$$\mathrm{CWA}_L = \{:M\neg P(x) \mid \neg P(x) \,/\, P \in \text{Pred and } x \text{ is a vector of variable symbols}\}.$$

A first order database over L is specified by a *positivist (first-order) default theory* $\Delta = (D, W)$, where W is a set of positive clauses and D a set of default rules satisfying the following properties:

1. $\mathrm{CWA}_L \subseteq D$, and
2. each default rule in $D - \mathrm{CWA}_L$ is of the form:

$$P_1(x_1) \wedge \cdots \wedge P_p(x_p) : M\neg Q_1(y_1), \ldots, M\neg Q_q(y_q) \mid R_1(z_1) \vee \cdots \vee R_r(z_r) \text{ with } p \geqslant 0,\ q > 0,\ r > 0 \text{ and}$$

$x_1, \ldots, x_p, y_1, \ldots, y_q, z_1, \ldots, z_r$ are vectors of terms.

In the definition above 1 states the CWA and 2 generalizes in a straightforward manner the definition of a (propositional) positivist default rule.

In Reiter (1980), the notion of extension is defined properly for the class of closed default theories. Thus here, in order to define the semantics of first-order databases, that is, in order to define the concept of extension for the class of positivist first-order default theories, we need to introduce the notion of ground instance of a positivist (first-order) default theory.

Given a clause α and x the set of variables occurring in α, a ground instance of α is a formula obtained by applying η to α, where η is a substitution mapping the set of variables x to a set of ground terms. A ground instance of a positivist default rule d is defined in exactly the same manner. Let $\Delta = (D, W)$ be a positivist (first-order) default theory. The *ground instance of* Δ is the default theory (D', W), where $D' = \{d' / \exists d \in D,\ d'$ is a ground instance of $d\}$. Note that if FUN is a nonempty set of function symbols then the set of default rules of the ground instance of Δ may be infinite. The ground instance of Δ is a closed default theory.

DEFINITION 3.6. Let Δ be a positivist (first-order) default theory and Δ' its ground instance. Then a set E of formulae in L is an *extension of* Δ iff E is an extension of Δ'.

We have shown that extensions of positivist propositional default theories are complete. This result (Lemma 3.1) is generalized as follows:

LEMMA 3.9. *Let $\Delta = (D, W)$ be a positivist (first-order) default theory over L such that W is consistent. Let E be an extension of Δ. Then*:

$$\forall P(t) \in H_L, \qquad P(t) \in E \text{ or } \neg P(t) \in E.$$

The proof of Lemma 3.9 is totally similar to the proof of Lemma 3.1 and is omitted here. Note that an extension E of a positivist (first-order) default theory is not always complete: it may exist a sentence ω such that $\omega \notin E$ and $\neg\omega \notin E$.

From the preceding lemma, we immediately deduce that:

COROLLARY 3.10. *Let $\Delta = (D, W)$ be a positivist (first-order) default theory over L such that W is consistent. Let E be an extension of Δ. Then*:

E has a unique Herbrand model.

The notion of stratifiable positivist (first-order) default theory is simply obtained by considering in Definition 3.4 a partition of the set of predicate symbols instead of a partition of the set of propositions. This definition is not detailed here. We prefer to directly define locally stratifiable positivist (first-order) default theories. Local stratifiability has been first introduced in Przymusinski (1988) in order to enlarge the class of "permissible logic programs with negation."

DEFINITION 3.7 (Przymusinski, 1988). Let $\Delta = (D, W)$ be a positivist first-order default theory over L. Let (D', W) be the ground instance of Δ. Δ is *locally stratifiable* iff there exists a sequence (possibly infinite) $S = H_1, ..., H_i, ...$, called *Herbrand stratification for* Δ, such that $H_1, ..., H_i, ...$ is a partition of the Herbrand base H_L and satisfies:

(i) For each $P_1(a_1) \wedge \cdots \wedge P_p(a_p) : M \neg Q_1(b_1), ..., M \neg Q_q(b_q) \mid R_1(c_1) \vee \cdots \vee R_r(c_r))$ in D', there exists $k \geqslant 1$ such that:

$$\forall j \in [1 \cdots r],\ R_j(c_j) \in H_k,$$

$$\forall j \in [1 \cdots p],\ \exists 1 \leqslant i \leqslant k / P_j(a_j) \in H_i,$$

$$\forall j \in [1 \cdots q],\ \exists 1 \leqslant i < k / Q_j(b_j) \in H_i.$$

(ii) For each ground instance, $P_1(a_1) \wedge \cdots \wedge P_p(a_p) \rightarrow R_1(c_1) \vee \cdots \vee R_r(c_r)$ of a clause in W, there exists $k \geqslant 1$ such that:

$$\forall j \in [1 \cdots r],\ R_j(c_j) \in H_k,$$

$$\forall j \in [1 \cdots p],\ \exists 1 \leqslant i \leqslant k / P_j(a_j) \in H_i.$$

Clearly, stratifiability entails local stratifiability. Indeed, given $S = \text{Pred}_1, ..., \text{Pred}_s$, a stratification for Δ, it is easy to construct from S a Herbrand stratification $S_H = H_1, ..., H_s$ for Δ. It suffices to consider for $i = 1 \cdots s$, L_i to be the first-order language over $\text{Const} \cup \text{Fun} \cup \text{Var} \cup \text{Pred}_i$ and to take H_i equal to the Herbrand base of L_i. In general, local stratification does not imply stratification.

At the end of this section, we develop a short and informal discussion on the advantages and the weaknesses of local stratifiability. Example 4.1.3 presents a logic program that is locally stratifiable but not stratifiable. For the sake of the presentation, we now pursue the generalization of the results obtained in the preceding sections.

It is clear that given a stratifiable database Δ and a stratification S for Δ, S induces a partition of Δ. As in Definition 3.5, we can make use of this partition in order to give an alternative definition of the semantics of Δ. In order to define the notion of "extension with respect to a stratification" in the case where the Herbrand stratification may be an infinite sequence, it suffices to take for E the union of the extensions obtained for each stratum, i.e., in Definition 3.5, (i) is replaced by $E = \bigcup_{i=0 \cdots \infty} E_i$. We give below the generalization of Theorem 3.8, Theorem 3.6 and Corollary 3.7:

THEOREM 3.11. *Let Δ be a locally stratifiable positivist default theory. Then*:

(i) *There exists a set E of formulae in L such that E is an extension of Δ.*

Assume now that Δ is stratifiable and that S and S' are two distinct stratifications for Δ.

(ii) *E is an extension of Δ with respect to S iff E is an extension of Δ with respect to S', and*

(iii) *E is an extension of Δ with respect to S iff E is an extension of Δ (with respect to Reiter's definition).*

Sketch of Proof. Let $\Delta = (D, W)$ and consider the positivist default theory $\Delta' = (D', W')$, where D' (resp., W') is the ground instance of D (resp., W). It is easy to show: (*) given a set E of ground formulae in L, $Th(W \cup E)$ is an extension of Δ iff E is an extension of Δ'. By definition of local stratifiability; (**) Δ is locally stratifiable entails that Δ' is locally stratifiable.

Now let us view the Herbrand base H of L as a set of propositions. Let us denote L_H the propositional language over H. Of course, Δ' can be viewed as a positivist propositional default theory. (**) directly implies that the propositional default theory Δ' is stratifiable.

(i) Using Corollary 3.7, we deduce that there exists a set E of formulae in L_H such that E is an extension of the propositional default theory Δ'. It is clear that an extension E of the propositional default theory Δ' can be viewed as an extension of the first-order default theory Δ and thus by (*), we have that there exists a set E of formulae in L such that E is an extension of Δ.

The proof of (ii) and (iii) is done in the same way, using the fact that stratifiability implies local stratifiability, using the Herbrand stratifications induced respectively by S and S', and using Theorems 3.5 and 3.6. ∎

4. COMPARATIVE STUDIES

This section is devoted to a comparative study of several formalizations of the intended meaning of logical databases/logic programs.

The main part of the discussion focuses on a comparison between the default logic approach and the perfect model approach. The comparison leads to two main results. First, for (locally) stratifiable databases, these two approaches are equivalent. Second, in general, that is, for some nonstratifiable logical databases, default logic and perfect model do not behave in the same way. As a matter of fact, the perfect model approach fails to capture the intended meaning of simple and useful logic programs (that are not locally stratifiable) while the default logic approach behave satisfactorily for these programs.

The iterative fixed point approach (Apt *et al.*, 1988), the autoepistemic approach (Gelfond, 1987) and the stable model approach (Gelfond and Lifschitz, 1988) are also examined versus the default logic approach.

4.1. *Default Logic versus Perfect Model*

For the class of locally stratifiable databases, we are going to show the equivalence between the default logic approach and the perfect model approach (Przymusinski, 1988). This latter approach is based on a preference relation among minimal models.

In order to establish the equivalence between the default semantics and the perfect semantics for the class of stratifiable logical databases, we proceed to the following intermediate steps. First, we show that default models of stratifiable databases have an alternative definition strictly based on the pure notion of minimal model of Bossu and Siegel (1985) (a model M is less than a model N iff M is included in N) while making an explicit use of the stratification. Second, we use that perfect models of stratifiable databases have also an alternative definition based on the notion of minimal model while making an explicit use of the stratification. Note here that this last result has been conjectured without proof in Bidoit and Hull (1989) and is also embedded in the proof of the main result of Przymusinski (1988, Theorem 4).

Besides serving the equivalence proof between default and perfect models, these intermediate results lead to the following remark that, we claim, is not stated clearly in the literature:

Once it is known that a logical database is stratifiable, then the notion of minimal model defined in terms of inclusion between models is totally sufficient to define the intended meaning of the database. This implies that concepts such as extension of default theories, supported models, and preference relation (among minimal models) are unnecessarily added to the notion of minimal model. Of course, we do not claim that these concepts are inadequate for nonstratifiable logical databases nor do we claim that logical databases outside the class of stratifiable databases do not make sense. These last points are briefly discussed later in this section.

For the purpose of discussion, we first need to proceed to a minor syntactical harmonization. In the default logic framework, logical databases and logic programs are syntactically expressed by default theories. In the perfect model framework, they are expressed by sets of rules called P-clauses.

The notion of P-clause has been first introduced in (Bidoit and Hull, 1986) to syntactically distinguish deductive rules of the form $A \wedge \neg B \rightarrow C$ from deductive rules of the form $A \rightarrow B \vee C$ which are equivalent from the first-order logic point of view but not from the deductive database or logic programming points of view.

DEFINITION 4.1.1. A *positivist clause* (*P-clause*) over L is an expression of the form:

$$P_1 \wedge \cdots \wedge P_p \wedge \neg Q_1 \wedge \cdots \wedge \neg Q_q \rightarrow R_1 \vee \cdots \vee R_r,$$

where $p \geqslant 0$, $q \geqslant 0$, and $r \geqslant 1$, $P_1, ..., P_p$, $Q_1, ..., Q_q$, $R_1, ..., R_r$ are propositions of L. We will denote by $\text{Prem}(\sigma) = \{P_1, ..., P_p, Q_1, ..., Q_q\}$, $N\text{Prem}(\sigma) = \{Q_1, ..., Q_q\}$, and $\text{Cons}(\sigma) = \{R_1, ..., R_r\}$.

The correspondence between positivist default theories and sets of P-clauses is straightforward to establish. In the following, the default theory associated with the set of P-clauses Σ (resp. the set of P-clauses associated with the positivist default theory Δ) is denoted by $\Delta_\Sigma = (D_\Sigma, W_\Sigma)$ (resp. Σ_Δ). In the following, the term "logical database" designates a positivist default theory or its associated set of P-clauses depending on the framework.

Now, we provide a semantics of stratifiable databases that is expressed exclusively in terms of minimal model. Variants of the following definition can be found in (Apt *et al.*, 1988; Gelfond, 1987; Przymusinski, 1989; Van Gelder, 1988). (Definition 4.1.2 is slightly different from these variants because it is a non-"procedural" definition in the sense that it is not based on propagation strata after strata of minimals models.)

DEFINITION 4.1.2. Let Σ be a stratifiable set of P-clauses over the propositional language L. Let $S = \text{Prop}_1 \cdots \text{Prop}_s$ be a stratification for Σ and let M be an interpretation of L. Then M is a *minimal model* of Σ *with respect to the stratification* S iff

$\forall i \in [1 \cdots s]$, $M_i = M \mid L_{\rightarrow i}$ is a minimal model of $\Sigma_{\rightarrow i}$, where

$M \mid L_{\rightarrow i}$ denotes the restriction of M to the language $L_{\rightarrow i}$, and

$\Sigma_i = \{\sigma \mid \sigma \in \Sigma \text{ and } \text{Cons}(\sigma) \in L_i\}$.

We now show that default models of stratifiable databases are minimal models with respect to any stratification.

LEMMA 4.1.1. *Let DB be a stratifiable logical database over the propositional language L. Let S be a stratification for DB. Then:*

> *M is a default model for DB iff M is a minimal model of DB with respect to S.*

Proof. Let $\Delta_\Sigma = (D, W)$ and $S = \text{Prop}_1 \cdots \text{Prop}_s$ be a stratification for Δ_Σ.

CLAIM 4.1.1. *Let $M_1 \cdots M_s$ such that $\forall i \in [1 \cdots s]$, M_i is an interpretation of $L_{\to i}$. Then: $\forall i \in [1 \cdots s]$, M_i is a minimal model of $\Sigma_{\to i}$ iff $\forall i \in [1 \cdots s]$, M_i is a minimal model of U_i where:*

$$U_i = \bigcup_{j=1 \cdots i} (W_j \cup U'_j)$$

and

$$U'_j = \{\alpha \to \gamma \mid d \in D_j, \text{prer}(d) = \alpha, \text{cons}(d) = \gamma, \textit{and}$$
$$\forall \neg Q \in \text{just}(d), \neg Q \in M_{j-1}\}$$

assuming $M_0 = \varnothing$.

The proof of the claim can be found in Appendix C.

($\Rightarrow$) Let M be a minimal model of Σ with respect to the stratification S. Claim 4.1.1 directly implies that $\forall i \in [1 \cdots s]$, M_i is a minimal model of U_i.

Let $E = Th(\{\sigma \mid \sigma \in L \text{ and } M \models \sigma\})$. To show that E is an extension of $\Delta_\Sigma = (D, W)$, by Theorem 3.6, it suffices to show that E is an extension of Δ_Σ with respect to S. Thus using Corollary 3.4, it remains to prove that $E_i = E \mid L_{\to i} = Th(\{\sigma \mid \sigma \in L_{\to i} \text{ and } M_i \models \sigma\})$ is an extension of $(\text{CWA}_{\to i}, V_i)$, where $V_i = \bigcup_{j=1 \cdots i}(W_j \cup W'_j)$ and $W'_j = \{\alpha \to \gamma \mid d \in D_j, \text{prer}(d) = \alpha, \text{cons}(d) = \gamma, \text{ and } \forall \neg Q \in \text{just}(d), \neg Q \in E_{j-1}\}$. Clearly, $\forall i \in [1 \cdots s]$, $V_i = U_i$. Thus $\forall i \in [1 \cdots s]$, M_i is a minimal model of $V_i = U_i$. By Theorem 3.2, we conclude that $\forall i \in [1 \cdots s]$, E_i is an extension of $(\text{CWA}_{\to i}, V_i)$.

($\Leftarrow$) Exactly the same way. ▌

Now we review the concept of perfect model introduced in Przymusinski (1988). Given a set Σ of P-clauses, Przymusinski's approach (*op. cit.*) consists in deducing from the syntax of Σ a relation, called a *priority relation*, on the propositions of the language. Then, this relation is used to characterize the "good" models of Σ, called *preferable models*. Intuitively, the definition of the priority relation associated with Σ is based on the three following principles:

(a) consequents of a P-clause should have lower priority (for minimization) than negative premises of this P-clause,

(b) consequents of a P-clause should have priority not higher than positive premises of this P-clause, and

(c) propositions occurring in a consequence of a P-clause should have the same priority.

Formally, a directed graph is associated with a set Σ of P-clauses (Apt *et al.*, 1988). The vertices of the graph are the propositions of the language L and the directed edges are pairs $[X, Y]$ of propositions such that: $\exists \sigma \in \Sigma$, $X \in \mathrm{Prem}(\sigma)$ and $Y \in \mathrm{Cons}(\sigma)$. If $X \in N\mathrm{Prem}(\sigma)$, then the edge is called a *negative edge*. Using this graph, the relation of *higher priority* (Przymusinski, 1988), denoted $>_\Sigma$, induced by Σ, is defined on $\mathrm{Prop} \times \mathrm{Prop}$ as follows: $X >_\Sigma Y$ iff there exists a path in the directed graph associated with Σ from X to Y going through a negative edge. As previously indicated, this relation of higher priority $>_\Sigma$ is then used to "compare" the models of Σ. Among the possible minimal models of Σ, we prefer the ones which minimize higher priority propositions even at the price of enlarging the truth of propositions of lower priority. This leads to the following notion of preferability:

DEFINITION 4.1.3 (Przymusinski, 1988). Let Σ be a set of P-clauses over the language L. Let M and N be two models of Σ. M is *preferable* to N, denoted $M \leqslant_{\mathrm{pref}} N$, iff:

$$\forall X \in M - N,\ \exists Y \in N - M / Y >_\Sigma X.$$

M is a *perfect model* of Σ iff there exists no model distinct from M and preferable to M.

Every perfect model is minimal (Przymusinski, 1988).

EXAMPLE 4.1.1. We specify below the logical database of Example 3.1 using a set of P-clauses:

$$\Sigma = \{\text{WEEK\_DAY},\ \text{WEEK\_DAY} \wedge \neg\text{SICK\_JOHN} \rightarrow \text{WORK\_JOHN},$$
$$\text{WORK\_JOHN} \rightarrow \text{TEACH\_JOHN} \vee \text{MEETING\_JOHN}\}.$$

This set of P-clauses induces the following priority relation:

(i) $\text{SICK\_JOHN} >_\Sigma \text{WORK\_JOHN}$,

(ii) $\text{SICK\_JOHN} >_\Sigma \text{TEACH\_JOHN}$.

(iii) $\text{SICK\_JOHN} >_\Sigma \text{MEETING\_JOHN}$.

Here are the three minimal models of Σ:

$$M = \{\text{WEEK\_DAY, WORK\_JOHN, TEACH\_JOHN}\}$$
$$M' = \{\text{WEEK\_DAY, WORK\_JOHN, MEETING\_JOHN}\}$$
$$M'' = \{\text{WEEK\_DAY, SICK\_JOHN}\}.$$

From (i), (ii), and (iii), we deduce that $M <_{\text{pref}} M''$ and $M' <_{\text{pref}} M''$. In conclusion, M and M' are the only perfect models of Σ.

Let us now consider the logic program $\Sigma = \{\neg A \rightarrow B, \neg B \rightarrow A\}$. It induces that $B >_{\Sigma} A$, $A >_{\Sigma} B$ and that the minimal models $\{A\}$, $\{B\}$ of Σ are preferable one to each other ($\{A\} \leqslant_{\text{pref}} \{B\} \leqslant_{\text{pref}} \{A\}$). In other words, Σ is inconsistent in the sense that it has no preferable model. In fact, for sets of P-clauses, recursion on negation leads to problems similar to the ones discussed in Section 3 for positivist default theories. Thus recursion on negation is eliminated by considering stratifiable sets of P-clauses. A set of P-clauses is stratifiable iff its associated positivist default theory is stratifiable.

EXAMPLE 4.1.2. Let Δ be the positivist default theory of Example 3.1 and Σ be the set of P-clauses of Example 4.1.1. Obviously $\Delta = \Delta_{\Sigma}$ and $\Sigma = \Sigma_{\Delta}$ and, since Δ is stratifiable, so is Σ. Note that the perfect models M and M' of Σ are respectively the first-order models of the extensions E and E' of Δ. In this case, the perfect model semantics and the default logic semantics coincide.

Now we state that for stratifiable databases, minimal models with respect to a stratification coincide with perfect models. Recall that this result is conjectured in Bidoit and Hull (1989) and indirectly stated in Przymusinski (1988).

LEMMA 4.1.2. *Let DB be a stratifiable logical database over the propositional language L. Let S be a stratification for DB. Then:*

M is a perfect model forDB iff M is a minimal model of DB with respect to S.

A complete proof of Lemma 4.1.2 can be found in Bidoit and Froidevaux (1988b). We just sketch the proof below.

Sketch of Proof. ($\Rightarrow$) Let M be a model of Σ such that for $i \in [1 \cdots s]$, $M_i = M \mid L_{\rightarrow i}$ is a minimal model of $\Sigma_{\rightarrow i}$. To show that M is a perfect model of Σ, we prove by induction on i that M_i is a perfect model of $\Sigma_{\rightarrow i}$.

In the induction step, given that for $i \leqslant p$, M_i is a perfect model of $\Sigma_{\to i}$, assuming that there exists a model N of $\Sigma_{\to p+1}$ such that $N <_{\text{pref}} M_{p+1}$ leads to a contradiction.

($\Leftarrow$) Let M be a perfect model of Σ. To show that for $i \in [1 \cdots s]$, $M_i = M \mid L_{\to i}$ is a minimal model of $\Sigma_{\to i}$, we proceed by induction on the number s of strata of the stratification S.

Induction step. We assume that each stratifiable set Σ of P-clauses whose stratification S has s strata, $s \leqslant p$, satisfies: (*) M is a perfect model of Σ implies that for $i \in [1 \cdots s]$, $M_i = M \mid L_{\to i}$ is a minimal model of $\Sigma_{\to i}$.

Let us consider a set Σ of P-clauses with a stratification $S = L_1 \cdots L_{p+1}$. To show that Σ satisfies (*), note that, given M a perfect model of Σ, because perfect models are minimal models, $M = M_{P+1}$ is a minimal model of $\Sigma = \Sigma_{\to P+1}$. Thus, by induction hypothesis, it remains to show that $M \mid L_{\to p} = M_P$ is a perfect model of $\Sigma_{\to P}$. At that point, we assume that there exists N' model of $\Sigma_{\to P}$ such that $N' <_{\text{pref}} M_P$ and we construct a model N of $\Sigma_{\to P+1}$ such that $N' = N \mid L_{\to p}$ (the construction starts with N', model of $\Sigma_{\to P}$, which is "augmented" with propositions of L_{p+1} in order to obtain a model that satisfies $\Sigma_{\to P+1} = \Sigma_{\to P} \cup \Sigma_{P+1}$) and such that $N <_{\text{pref}} M$ (the propositions added to N' are carefully chosen). This leads to a contradiction. ▮

From the two preceding lemma, the equivalence of the default model semantics and the perfect model semantics for stratifiable databases is immediate:

THEOREM 4.1.3. *Let DB be a stratifiable logical database over the propositional language L. Then*:

M is a default model for DB iff M is a perfect model for DB.

Note that this equivalence is modular in the sense of Imielinski (1985). It is easy to check that updates (insertions of facts) performed on a set of P-clauses Σ and on its corresponding default theory Δ_Σ produce a set $\Sigma+$ of P-clauses and a default theory $(\Delta_\Sigma)+$ semantically equivalent in the sense that M is a perfect model of $\Sigma+$ iff M is the model of an extension E of $(\Delta_\Sigma)+$. In fact this result is straightforward, noticing that in the case of simple updates like insertion of facts, $(\Delta_\Sigma)+ = \Delta_{\Sigma+}$.

A syntactical correspondence between sets of (first-order) P-clauses and positivist (first-order) default theories is easy to establish. Definition 4.1.2 is generalized to the first order case (Przymusinski, 1988) by assuming M and N to be Herbrand models. The generalization of Lemma 4.1.2 is immediate and thus:

THEOREM 4.1.4. *Let Σ be a set of P-clauses. If Σ is locally stratifiable, then M is a perfect model of Σ iff M is a default model of the positivist default theory Δ_{Σ}.*

The proof of the above result is obvious using the generalization of Lemma 4.1.2 and proceeding as in the proof of Theorem 3.11.

The last part of the comparison between the perfect model approach and the default approach is devoted to an informal discussion on the advantages and the weaknesses of local stratifiability and also on the inadequacy of the perfect model approach.

We start by presenting an example of a nonstratifiable program which is locally stratifiable. This example due to V. Lifschitz has also been presented in Przymusinski (1988).

EXAMPLE 4.1.3. The following progam gives a definition of even natural numbers. Using the default logic approach, we have:

PROGRAM 4.3

$$\underbrace{\begin{array}{l} :M\,\neg\mathrm{EVEN}(x) \mid \neg\mathrm{EVEN}(x) \\ :M\,\neg\mathrm{EVEN}(x) \mid \mathrm{EVEN}(\mathrm{suc}(x)) \end{array}}_{D} \qquad \underbrace{\mathrm{EVEN}(0)}_{W}$$

The function symbol suc denotes the successor function. (D, W) is not stratifiable but it is locally stratifiable. The Herbrand stratification for (D, W) is $S = (H_i)_{i \geqslant 0}$, with $H_i = \{\mathrm{EVEN}(\mathrm{suc}^i(0))\}$ and $\mathrm{suc}^0(0) = 0$, $\mathrm{suc}^i(0) = \mathrm{suc}(\mathrm{suc}^{i-1}(0))$. Note here that the Herbrand stratification S for (D, W) is an infinite sequence.

We now propose a second example of a nonstratifiable program which is not, this time, locally stratifiable. However, the following program is, in its spirit, very similar to the one given in Example 4.1.3.

EXAMPLE 4.1.4. The following program describes a branch of a genealogical tree, having only three individuals a, b, and c. These individuals inherit of the property P every other generation (starting from a). Using the default logic approach, this program is given by:

PROGRAM 4.4

$$\underbrace{\begin{array}{l} :M\,\neg\mathrm{FATHER}(x, y) \mid \neg\mathrm{FATHER}(x, y) \\ :M\,\neg P(x) \mid \neg P(x) \\ \mathrm{FATHER}(x, y) :M\,\neg P(x) \mid P(y) \end{array}}_{D} \qquad \underbrace{P(a),\ \mathrm{FATHER}(a, b),\ \mathrm{FATHER}(b, c)}_{W}$$

First note that the language used to write the Program 4.4 does not contain any function symbols. It is easy to establish a correspondence between Program 4.3 and Program 4.4: the predicate FATHER (resp., the predicate P) of Example 4.1.4 plays the same role as the function suc (resp., the predicate EVEN) of Example 4.1.3.

Because the unquestionable similarities of Programs 4.3 and 4.4:

(1) there is no reason to reject Program 4.4. Unfortunately, it is straightforward to check that Program 4.4 is not stratifiable. Neither it is locally stratifiable.

(2) it is beyond argument that the semantics of Program 4.4 is as clear as the semantics of Program 4.3. Unfortunately, the set of P-clauses specifying Program 4.4 has **no perfect model**. On the other hand, the default theory specifying Program 4.4 and given above **has** the following expected **unique extension**:

$$\begin{aligned} E = Th(W \cup \{&\text{FATHER}(a, b), && \text{FATHER}(b, c), && \neg\text{FATHER}(a, a) \\ &\neg\text{FATHER}(a, c), && \neg\text{FATHER}(b, a), && \neg\text{FATHER}(b, b), \\ &\neg\text{FATHER}(c, a), && \neg\text{FATHER}(c, b), && \neg\text{FATHER}(c, c), \\ &P(a), && \neg P(b), && P(c)\}). \end{aligned}$$

In conclusion, we have: (1) Program 4.4 is not locally stratifiable and thus should be rejected, which is counter-intuitive. Moreover, (2) according to the perfect model approach, Program 4.4 is "inconsistent."

Conclusion (1) above implies that local stratifiability is still too restrictive. The refinement of stratifiability consisting in a syntactical analysis of the "complete" instantiation of the program is too naive in general. One could relax the stratification constraint or/and think of other reasonable methods to check whether a program is acceptable. Further work in this direction has been presented in (Bidoit and Froidevaux, 1988a, 1988b).

Conclusion (2) above has a more fundamental impact. Indeed, according to the perfect model approach, in order to make Program 4.4 permissible, it would be necessary not only to relax local stratifiability but also to modify the definition of a perfect model (see Przymusinska and Przymusinski, 1988). In a context where the notion of acceptable (stratifiable) program and the notion of consistent program (program having at least one perfect model) do not completely merge anyway, this process of generalization reveals some imperfection of the perfect model approach. Whereas, according to the default logic approach, in order to make Program 4.4 permissible, it suffices to relax local stratifiability. Our claim is that the perfect model approach is strongly dependent on a **local syntactical** analysis of the program. Thus the semantics of negation induced by this approach is locally determined for each program. As a matter of fact, because of this, it is very likely that the generalization of stratifiability and the generalization of perfect model could be done at the same time. On the other hand, the default logic approach is rather based on a **global interpretation** of negation: negation is interpreted as "negation as default" no matter where and how it occurs in a program.

We would like to terminate the comparison by showing with a last example that the default logic approach and the perfect model approach diverge for nonlocally stratifiable programs and also that once again the perfect model approach seems to have an awkward behavior with respect to disjunction.

EXAMPLE 4.1.5. Consider the program specified by the positivist default theory $\Delta = (D, W)$, where $D = \mathrm{CWA}_L \cup \{:M \neg A/B, :M \neg B/C, :M \neg C/A \vee D\}$ and $W = \emptyset$. This program (which is not stratifiable) has one extension, namely $\{\neg A, B, \neg C, D\}$. Note that the corresponding set Σ of P-clauses has no perfect model.

Let us now split the program in two parts $\Delta_1 = (D_1, W_1)$ and $\Delta_2 = (D_2, W_2)$, where

$$D_1 = \mathrm{CWA}_L \cup \{:M \neg A/B, :M \neg B/C, :M \neg C/A\}, \qquad W_1 = \emptyset,$$

$$D_2 = \mathrm{CWA}_L \cup \{:M \neg A/B, :M \neg B/C, :M \neg C/D\}, \qquad W_2 = \emptyset.$$

The program Δ_1 has no extension but the program Δ_2 has one extension namely $\{\neg A, B, \neg C, D\}$. Note that the set of extensions for Δ is the union of the set of extensions for Δ_1 and the set of extensions for Δ_2.

Let Σ_1 (resp., Σ_2) be the set of P-clauses corresponding to Δ_1 (resp., Δ_2). It is interesting to note that Σ_1 has no perfect model but Σ_2 has a unique perfect model which is $\{B, D\}$ and thus the set of perfect models for Σ is not equal to but included in the union of the set of perfect models for Σ_1 and the set of perfect models for Σ_2.

4.2. *Other Comparisons*

In Przymusinski (1988), it is shown that for stratifiable logic programs (no disjunction),

(i) they have a unique perfect model;

(ii) this unique perfect model coincides with the model obtained by the iterative fixed point method Apt *et al.* (1988) and by Van Gelder's (1988) method;

(iii) perfect models can also be characterized using the concept of prioritized circumscription.

In Lifschitz (1988), the equivalence between pointwise circumscription and iterative fixed point is emphasized. From the above-mentioned equivalence and Theorem 4.1.3 stating the equivalence of the default semantics and the perfect model semantics for stratifiable databases, we directly infer that:

COROLLARY 4.2.1. *Default logic, perfect model, iterative fixed point, and (pointwise) circumscription associate the same meaning to stratifiable logic programs.*

(a) *Iterative fixed point and minimal model.* Let us briefly show how a slightly different definition of stratification implies that the minimal model with respect to stratification is not sufficient to define the declarative semantics of stratifiable databases.

In Apt *et al.* (1988), the semantics of logic programs is defined using iterative fixed point. An alternative definition is given that uses the notions of minimal model *and* supported model. The notion of supported model, simultaneously introduced in Bidoit and Hull (1986, 1989) and called there the justified model, can be intuitively defined as follows: given a set of P-clauses Σ and a model M of Σ, a fact A true for M is supported (or justified) by Σ iff A is the consequence of some P-clause in Σ whose premises are all true for M. M is a supported (or justified) model of Σ iff each positive true fact in M is supported (or justified) by Σ. For example, let us consider the set $\Sigma = \{\neg q \rightarrow p\}$ of P-clauses. Viewed as a set of clauses, Σ has two minimal models, namely $M_1 = \{p\}$ and $M_2 = \{q\}$. M_1 is a supported model of Σ but M_2 is not a supported model of Σ because there exists no P-clause in Σ with q as consequent.

Now in Apt *et al.* (1988), a logic program Σ is stratifiable iff there exists a partition $\Sigma_1, ..., \Sigma_s$ of Σ such that if P is a positive premise of a clause in Σ_i then its "definition" is in $\Sigma_{\rightarrow i}$ and if P occurs negatively in the premises of a clause in Σ_i then its "definition" is in $\Sigma_{\rightarrow i-1}$ (Σ_1 can be empty). The semantics of a stratifiable logic program is then defined as follows: M is a "good" model of Σ iff $M = M_s$, where for $i = 1 \cdots s$, M_i is a minimal *and* supported model of $\Sigma_i \cup P_{i-1}$ (where P_i is the set of positive facts true for M_i—let us assume $P_0 = \varnothing$). Let us go back to the logic program $\Sigma = (\neg q \rightarrow p\}$:

Apt *et al.* (1988) approach: Σ is stratifiable and $\Sigma_1 = \Sigma$ is a stratification for Σ. Σ_1 has two minimal models $M_1 = \{p\}$ and $M'_1 = \{q\}$. M_1 is a supported model of $\Sigma_1 = \Sigma$ thus a "good" model of Σ. M'_1 is not. The notion of supported model *is needed* in Apt *et al.* (1990) in order to eliminate the minimal model M'_1.

Bidoit and Froidevaux approach: Σ is stratifiable and $\text{Prop}_1 = \{q\}$, $\text{Prop}_2 = \{p\}$ is a stratification for Σ. It induces the following partition of Σ: $\Sigma_1 = \emptyset$ and $\Sigma_2 = \Sigma = \{\neg q \rightarrow p\}$. Σ_1 has a unique minimal model, namely $M_1 = \emptyset$. $M_2 = \{p\}$ is the unique minimal model of Σ_2 such that its restriction to L_1 is M_1. (One does not need here to use the notion of supported model in order to eliminate the minimal model $M'_2 = \{p\}$ of Σ_2; we use the minimal model M_1 of the preceding stratum Σ_1.)

(b) *Autoepistemic logic.* Autoepistemic logic has been introduced by Moore (1985) in order to formalize a type of non-monotonic reasoning, called autoepistemic reasoning. In this framework, the language is augmented by a modal operator L and $L\alpha$ means that the formula α is one of the agent's beliefs. As in default logic, a fixed point definition is used for describing the sets of theorems, called *stable expansions*, associated with an autoepistemic theory.

DEFINITION 4.2.1. Let T be an autoepistemic theory. Then a set of theorems E is a stable expansion for T iff E satisfies

$$E = Th(T \cup \{LP/P \in E\} \cup \{\neg LP/P \notin E\}).$$

Gelfond (1987) defines and studies a specific class of autoepistemic theories. In particular, the relationship between logic programming and autoepistemic logic is drawn by means of a suitable translation of logic programs into autoepistemic theories. With the P-clause $P_1 \wedge \cdots \wedge P_p \wedge \neg Q_1 \wedge \cdots \wedge \neg Q_q \rightarrow R$ is associated the autoepistemic sentence $P_1 \wedge \cdots \wedge P_p \wedge \neg LQ_1 \wedge \cdots \wedge \neg LQ_q \rightarrow R$. Gelfond is naturally driven to consider stratifiable programs and establishes the following results.

Result (Gelfond, 1987). For any stratifiable logic program P, the associated autoepistemic theory has a unique stable expansion E and, for each proposition A in Prop, $A \in E$ iff $A \in M$, and $\neg LA \in E$ iff $A \notin M$, where M is the canonical model defined by Apt *et al.* (1988).

The equivalence between autoepistemic semantics and iterative fixed point semantics established in Gelfond (1987) for stratifiable logic programs and the equivalence between iterative fixed point semantics and default semantics (Corollary 4.2.1) obtained above through the perfect model semantics lead to:

COROLLARY 4.2.2. *Let P be a stratifiable logic program. Let E be the unique extension of the positivist default theory associated with P. Let E' be the unique stable expansion of the autoepistemic theory associated with P.*

Then:

For each proposition A in Prop, *$A \in E$ iff $A \in E'$, and $\neg A \in E$ iff $\neg LA \in E'$.*

The following example illustrates the notion of autoepistemic program and stable expansion.

EXAMPLE 4.2.1. Let P be the stratifiable logic program specified by the following set of P-clauses:

$$P = \{A \wedge \neg B \rightarrow C, \neg C \rightarrow D\}.$$

The corresponding autoepistemic theory is given by

$$T = \{A \wedge \neg LB \rightarrow C, \neg LC \rightarrow D\};$$

T has a unique stable expansion $E' = Th(\{\neg LA, \neg LB, \neg LC, LD, D\})$. Finally, following the default approach, P is specified by the positivist default theory $\Delta = (D, \emptyset)$, where

$$D = \{:M \neg A/\neg A, :M \neg B/\neg B, :M \neg C/\neg C, :M \neg D/\neg D,$$
$$A :M \neg B/C, :M \neg C/D\};$$

Δ has a unique extension $E = Th(\{\neg A, \neg B, \neg C, D\})$.

Although the default approach and the autoepistemic approach coincide for the class of stratifiable programs (no disjunction), they behave differently in general. As a matter of fact, for stratifiable logical databases (disjunction is allowed), the two approaches are not equivalent as is shown by the next example.

EXAMPLE 4.2.2. Consider the stratifiable logical database specified by the set of P-clauses

$$DB = \{A \vee B, A \wedge \neg C \rightarrow D, B \wedge \neg C \rightarrow D\}.$$

The corresponding autoepistemic theory is given by

$$T = \{A \vee B, A \wedge \neg LC \rightarrow D, B \wedge \neg LC \rightarrow D\};$$

T has a unique stable expansion $\{A \vee B, L(A \vee B), \neg LA, \neg LB, \neg LC, D, LD\}$. Using the same procedure as in the result of Gelfond (1987) in order to establish a correspondence between expansions and standard models, we would have that, because $\neg LA$ (resp., $\neg LB$) is in the expansion, $\neg A$ (resp. $\neg B$) is in the corresponding model. Let M be this model, M should satisfy $A \vee B$, which is contradictory with A and B false for M.

Following the default logic approach, the logical database is expressed by the positivist default theory $\Delta = (D, \{A \vee B\})$, where

$$D = \{:M \neg A/\neg A,\ :M \neg B/\neg B,\ :M \neg C/\neg C,\ :M \neg D/\neg D,$$
$$A :M \neg C/D,\ B :M \neg C/D\};$$

Δ has two extensions, $E_1 = Th(\{A, \neg B, \neg C, D\})$ and $E_2 = Th(\{\neg A, B, \neg C, D\})$. Because of the already established equivalence of the perfect model approach and the default approach, it is not surprising to note that $M_1 = \{A, D\}$ and $M_2 = \{B, D\}$ are the two perfect models for DB.

The reason why the autoepistemic approach and the default approach diverge in this case can be found in Konolige (1987), where a translation between both logics is investigated. In fact the autoepistemic theory T proposed by Gelfond (1987) in order to specify the database DB is not a "good" translation of Δ. A translation of Δ into a "quasi-equivalent" autoepistemic theory is given by

$$T' = \{\neg LA \rightarrow \neg A,\ \neg LB \rightarrow \neg B,\ \neg LC \rightarrow \neg C,\ \neg LD \rightarrow \neg D,$$
$$A \vee B,\ LA \wedge \neg LC \rightarrow D,\ LB \wedge \neg LC \rightarrow D\}$$

Now Δ and T' are said to be quasi-equivalent (Konolige, 1987) because: E is an extension of Δ iff E is the kernel (i.e., subset of formulae without modal operator) of a minimal stable expansion of T'.

The autoepistemic theory T' above has two minimal stable expansions that are:

$E'_1 = Th(\{A, LA, \neg B, \neg LB, \neg C, \neg LC, D, LD\})$ whose kernel is $Th(\{A, \neg B, \neg C, D\}) = E_1$;

$E'_2 = Th(\{\neg A, \neg LA, B, LB, \neg C, \neg LC, D, LD\})$ whose kernel is $Th(\{\neg A, B, \neg C, D\}) = E_2$.

Note that the four first sentences of T' resemble the CWA-default rules of Δ. However, the introduction of these sentences in the specification of DB in terms of autoepistemic theory is not essential (because the occurrence of $\neg LA$ was interpreted as having $\neg A$). On the other hand, the introduction of the modal operator L in front of positive premises of rules is decisive.

(c) *Stable model.* Finally, we propose a quick comparison of the default approach with the stable model approach (Gelfond and Lifschitz, 1988). Stable models have been introduced in *op. cit.* as a new semantics for logic programs. Stable models date from much later than default models and are nothing else than default models. In Bidoit and Froidevaux (1988b) we prove that:

THEOREM 4.2.3. *Let P be a logic program*:

M is a stable model for P iff M is a default model for P.

We insist here on the fact that in this result, the logic program P is not assumed to be stratifiable.

APPENDIX A

CLAIM 3.1. *Let $E_1, ..., E_s$ be a sequence such that for $i = 1 \cdots s$, E_i is a set of sentences of $L_{\rightarrow i}$. Then, $E_1, ..., E_s$ satisfies condition* (ii) *of Definition* 3.5 *iff $E_1, ..., E_s$ satisfies condition* (ii) *of Corollary* 3.4.

Proof of Claim 3.1. We proceed by induction on i:

i = 1. $D_1 = \text{CWA}_1$, thus $W'_1 = \varnothing$ and it directly follows that E_1 is an extension of (D_1, W_1) iff it is an extension of $(\text{CWA}_{\rightarrow 1}, V_1)$.

Induction step. Assume now that for each $i \leqslant k \leqslant s$, E_i is an extension of $(D_i, W_i \cup E_{i-1})$ iff E_i is an extension of $(\text{CWA}_{\rightarrow i}, V_i)$. Let E_{k+1} be an extension of $(D_{k+1}, W_{k+1} \cup E_k)$. Since S is a stratification for Δ, by Lemma 3.3: E_{k+1} is an extension of $(D_{k+1}, W_{k+1} \cup E_k)$ iff E_{k+1} is an extension of $(\text{CWA}_{\rightarrow k+1}, W_{k+1} \cup W'_{k+1} \cup E_k)$ iff by Theorem 3.2b: $E_{k+1} = Th(W_{k+1} \cup W'_{k+1} \cup E_k \cup \{\neg Q \mid \neg Q \in E_{k+1}\})$.

By induction hypothesis, E_k is an extension of $(\text{CWA}_{\rightarrow k}, V_k)$, and by Theorem 3.2b;

$$E_k = Th\left(\bigcup_{j=1 \cdots k} (W_j \cup W'_j) \cup \{\neg Q \mid \neg Q \in E_k\}\right).$$

Since $\{\neg Q \mid \neg Q \in E_k\} \subseteq \{\neg Q \mid \neg Q \in E_{k+1}\}$, it follows that

$$\begin{aligned} E_{k+1} &= Th\left(\bigcup_{j=1 \cdots k+1} (W_j \cup W'_j) \cup \{\neg Q \mid \neg Q \in E_{k+1}\}\right) \\ &= Th(V_{k+1} \cup \{\neg Q \mid \neg Q \in E_{k+1}\}). \end{aligned}$$

Using Theorem 3.2b once again, we conclude that E_{k+1} is an extension of $(D_{k+1}, W_{k+1} \cup E_k)$ iff E_{k+1} is an extension of $(\text{CWA}_{\rightarrow k+1}, V_{k+1})$. ▮

APPENDIX B

CLAIM 3.2. *Let $S = \mathrm{Prop}_1 \cdots \mathrm{Prop}_s$ be a stratification for Δ and E be an extension of Δ. Then $\forall i \in [1 \cdots \infty[$, $\forall j \in [1 \cdots s]$, $Th(G_i) \mid L_{\to j} = Th(G_i \mid L_{\to j})$, where*

$$E = \bigcup_{k=0 \cdots \infty} G_k \text{ with } G_0 = W, \text{ and } G_{k+1} = Th(G_k) \cup T_k \text{ and}$$

$$T_k = \{\mathrm{cons}(d) \mid d \in D,\ \mathrm{prer}(d) \in G_k,\ \forall(\neg Q) \in \mathrm{just}(d),\ \neg Q \in E\}.$$

Proof of Claim 3.2. 1. $\forall i \in [1 \cdots \infty[$, $\forall j \in [1 \cdots s]$, $G_i \mid L_{\to j} \subseteq G_i$ and thus $\forall i \in [1 \cdots \infty[$, $\forall j \in [1 \cdots s]$, $Th(G_i \mid L_{\to j}) \subseteq Th(G_i) \mid L_{\to j}$.

2. It remains to show that $\forall i \in [1 \cdots \infty[$, $\forall j \in [1 \cdots s]$, $Th(G_i) \mid L_{\to j} \subseteq Th(G_i \mid L_{\to j})$. For $k = 0 \cdots \infty$ let $T''_k = \{\mathrm{cons}(d) \mid d \in D\text{-CWA}_L,\ \mathrm{prer}(d) \in G_k,\ \forall(\neg Q) \in \mathrm{just}(d),\ \neg Q \in E\}$ and $T'_0 = \{\neg P / P \in \mathrm{Prop} \text{ and } \neg P \in E\}$. Then $G_k = Th(W \cup T'_0 \cup T''_{\to k-2}) \cup T''_{k-1}$ and $Th(G_k) = Th(W \cup T'_0 \cup T''_{\to k-1})$.

In the following, we show that the existence of a linear refutation in $W \cup T'_0 \cup T''_{\to k-1}$ for a clause $\alpha \in L_{\to j}$ implies the existence in $(W \cup T'_0 \cup T''_{\to k-1}) \mid L_{\to j}$ of a linear refutation for α.

A linear refutation r in V for α is described by a sequence $r = \langle (b_1, c_1) \cdots (b_p, c_p) \rangle$, where:

i. $b_1 = \alpha$, resolvant(b_p, c_p) is the empty clause,

ii. for $i = 1 \cdots p-1$, b_{i+1} is the resolvant of b_i and c_i, and

iii. for $i = 1 \cdots p$, either $c_i \in V$ or $c_i = b_j$ for some $j < i$.

We proceed by induction on the length $|r|$ of the refutation r in $(W \cup T'_0 \cup T''_{\to k-1})$ for $\alpha \in L_{\to j}$:

$|r| = 1$. By definition, r is a refutation in $(W \cup T'_0 \cup T''_{\to k-1}) \mid L_{\to j}$ for α.

Induction step. Assume that if r is a refutation in $(W \cup T'_0 \cup T''_{\to k-1})$ for $\alpha \in L_{\to j}$ and $|r| \leqslant p$ then there exists a refutation in $(W \cup T'_0 \cup T''_{\to k-1}) \mid L_{\to j}$ for α. Consider a refutation $r = \langle (b_1, c_1) \cdots (b_{p+1}, c_{p+1}) \rangle$ in $(W \cup T'_0 \cup T''_{\to k-1})$ for α. By definition of a refutation, $c_1 \in (W \cup T'_0 \cup T''_{\to k-1})$. Two cases arise:

$c_1 \in (T'_0 \cup T''_{\to k-1})$. Then clearly because S is a stratification for Δ and $\alpha \in L_{\to j}$, $c_1 \in L_{\to j}$. Hence $b_2 \in L_{\to j}$. $\langle (b_2, c_2) \cdots (b_{p+1}, c_{p+1}) \rangle$ is a refutation in $(W \cup T'_0 \cup T''_{\to k-1})$ for $b_2 \in L_{\to j}$ and then by induction hypothesis there exists a refutation in $(W \cup T'_0 \cup T''_{\to k-1}) \mid L_{\to j}$ for b_2 and thus a refutation in $(W \cup T'_0 \cup T''_{\to k-1}) \mid L_{\to j}$ for α.

$c_1 \in W$. Let $\alpha = A_1 \wedge \cdots \wedge A_m \to B_1 \vee \cdots \vee B_{m'}$.

$m=0$. $\alpha=(B_1 \vee \cdots \vee B_{m'})$ and the existence of a refutation r in $(W \cup T'_0 \cup T''_{\to k-1})$ for α implies that $\neg\alpha=(\neg B_1 \wedge \cdots \wedge \neg B_{m'}) \in Th(W \cup T'_0 \cup T''_{\to k-1})$ and thus obviously $\forall i \in [1 \cdots m'] \neg B_i \in T'_0$ and by hypothesis $B_i \in L_{\to j}$, thus there exists a refutation in $(W \cup T'_0 \cup T''_{\to k-1}) \mid L_{\to j}$ for α.

$m>0$. Let $l \in [1 \cdots p+1]$ be such that $l=\min\{i / A_1$ occurs in b_i not in $b_{i+1}\}$. Then clearly, $c_l \in (W \cup T'_0 \cup T''_{\to k-1})$. Consider $r'=\langle (b'_1, c'_1) \cdots (b'_{p+1}, c'_{p+1}) \rangle$ defined by:

$b'_1=\alpha,\ c'_1=c_l,$

for $i=2 \cdots l,\ c'_i=c_{i-1},$

for $i=l+1 \cdots p+1,\ c'_i=c_i$, and,

for $i=1 \cdots p,\ b'_{i+1}=\text{resolvant}(b'_i, c'_i)$.

It is easy to verify that r' is a refutation in $(W \cup T'_0 \cup T''_{\to k-1})$ for α. Now, because S is a stratification for Δ and $A_1 \in \text{Prop}_{\to j}$, we deduce that $c'_l \in L_{\to j}$.

Thus $b'_2=\text{resolvant}(b'_1, c'_1) \in L_{\to j}$ and by induction hypothesis, because $\langle (b'_2, c'_2) \cdots (b'_{p+1}, c'_{p+1}) \rangle$ is a refutation in $(W \cup T'_0 \cup T''_{\to k-1})$ for b_2, there exists a refutation in $(W \cup T'_0 \cup T''_{\to k-1}) \mid L_{\to j}$ for b'_2 and thus there exists a refutation in $(W \cup T'_0 \cup T''_{\to k-1}) \mid L_{\to j}$ for α. ▮

Appendix C

Claim 4.1. *Let $\Delta_\Sigma=(D, W)$ and $S=\text{Prop}_1 \cdots \text{Prop}_s$ be a stratification for Δ_Σ. Let $M_1 \cdots M_s$ be such that $\forall i \in [1 \cdots s]$, M_i is an interpretation of $L_{\to i}$. Then: $\forall i \in [1 \cdots s]$, M_i is a minimal model of $\Sigma_{\to i}$ iff $\forall i \in [1 \cdots s]$, M_i is a minimal model of U_i, where*

$U_i=\bigcup_{j=1 \cdots i}(W_j \cup U'_j)$ and

$U'_j=\{\alpha \to \gamma \mid d \in D_j,\ \text{pred}(d)=\alpha,\ \text{cons}(d)=\gamma,$ and $\forall \neg Q \in \text{just}(d), \neg Q \in M_{j-1}\}$ assuming $M_0=\varnothing$.

Proof of Claim 4.1. We proceed by induction on i.

i = 1. It is clear that $\Sigma_1=W_1$ and $U'_1=\varnothing$. Thus M_1 is a minimal model of Σ_1 iff M_1 is a minimal model of U_1.

Induction step. Suppose now that for $i \leqslant k < s$, M_i is a minimal model of $\Sigma_{\to i}$ iff M_i is a minimal model of U_i. Let us show that M_{k+1} is a minimal model of $\Sigma_{\to k+1}$ iff M_{k+1} is a minimal model of U_{k+1}.

We first show that a set M' of propositions such that $M' \mid L_{\to k} = M_k$ is a model of $\Sigma_{\to k+1}$ iff it is a model of U_{k+1}. Assume M' is a model of $\Sigma_{\to k+1}$. By induction hypothesis, $M' \mid L_{\to k} = M_k$ is a minimal model of $\Sigma_{\to k}$ and of U_k. Thus in order to show that M' is a model of $U_{k+1} = U_k \cup W_{k+1} \cup U'_{k+1}$, it remains to prove that M' satisfies $W_{k+1} \cup U'_{k+1}$. Since $W_{k+1} \subseteq \Sigma_{\to k+1}$ and M' is model of $\Sigma_{\to k+1}$, M' satisfies W_{k+1}.

Now consider $\sigma \in U'_{k+1}$. Then $\exists d \in D_{k+1}$ such that $\forall \neg Q \in \text{just}(d)$, $Q \notin M_k$ (thus $Q \notin M_{k+1}$) and $\sigma = \text{prer}(d) \to \text{cons}(d)$. Let $\text{just}(d) = \{\neg Q_1, ..., \neg Q_q\}$. Then, $\sigma' = \text{prer}(d) \wedge \neg Q_1 \wedge \cdots \wedge \neg Q_q \to \text{cons}(d) \in \Sigma_{k+1}$. Since M' is a model of $\Sigma_{\to k+1}$, M' satisfies σ' and thus σ.

Assume now that M' is a model of U_{k+1}. By induction hypothesis, $M' \mid L_{\to k} = M_k$ is a minimal model of U_k and of $\Sigma_{\to k}$. Thus in order to show that M' is a model of $\Sigma_{\to k+1} = \Sigma_{\to k} \cup \Sigma_{k+1}$, it remains to prove that M' satisfies

$$\Sigma_{k+1} = W_{k+1} \cup \{\text{prer}(d) \wedge \neg Q_1 \wedge \cdots \wedge \neg Q_q \to \text{cons}(d) \mid d \in D_{k+1} \text{ and } \\ \text{just}(d) = \{\neg Q_1, ..., \neg Q_q\}\}.$$

Since $W_{k+1} \subseteq U_{k+1}$ and M' is model of U_{k+1}, M' satisfies W_{k+1}. Consider $\sigma \in \Sigma_{k+1} - W_{k+1}$. Then $\sigma = \text{prer}(d) \wedge \neg Q_1 \wedge \cdots \wedge \neg Q_q \to \text{cons}(d)$ with $d \in D_{k+1}$ and $\text{just}(d) = \{\neg Q_1, ..., \neg Q_q\}$. Assume M' satisfies $\text{prer}(d) \wedge \neg Q_1 \wedge \cdots \wedge \neg Q_q$ then $\forall i \in [1 \cdots q]$, $Q_i \notin M_k$ and $\text{prer}(d) \to \text{cons}(d) \in U'_{k+1}$. Since M' is model of U_{k+1}, M' satisfies $\text{prer}(d) \to \text{cons}(d)$ and thus σ.

Assuming that M_{k+1} is a minimal model of $\Sigma_{\to k+1}$, let us show that M_{k+1} is a minimal model of U_{k+1}. In order to do this, assume that there exists a model M' of U_{k+1} such that $M' < M_{k+1}$. Then by definition, there exists $P \in L_{\to k+1}$ such that $P \notin M'$ and $P \in M_{k+1}$. Since $M' \mid L_{\to k}$ is a model of U_k and by induction hypothesis, M_k is a minimal model of U_k, $M' \mid L_{\to k} < M_{k+1} \mid L_{\to k} = M_k$ implies $M' \mid L_{\to k} = M_k$. We can deduce now that M' is a model of $\Sigma_{\to k+1}$ and $M' < M_{k+1}$, a contradiction.

Assuming that M_{k+1} is a minimal model of U_{k+1}, let us show that M_{k+1} is a minimal model of $\Sigma_{\to k+1}$. In order to do this, assume that there exists a model M' of $\Sigma_{\to k+1}$ such that $M' < M_{k+1}$. Then by definition, there exists $P \in \text{Prop}_{\to k+1}$ such that $P \notin M'$ and $P \in M_{k+1}$. Since $M' \mid L_{\to k}$ is a model of $\Sigma_{\to k}$ and by induction hypothesis, M_k is a minimal model of $\Sigma_{\to k}$, $M' \mid L_{\to k} < M_{k+1} \mid L_{\to k} = M_k$ implies $M' \mid L_{\to k} = M_k$. We can now deduce that M' is a model of U_{k+1} and $M' < M_{k+1}$, a contradiction. ∎

RECEIVED October 19, 1987; FINAL MANUSCRIPT RECEIVED September 14, 1989

REFERENCES

APT, R. K., BLAIR, H., AND WALKER, A. (1988), Towards a theory of declarative knowledge, *in* "Foundations of Deductive Databases and Logic Programming" (J. Minker, Ed.), pp. 89–148, Morgan Kaufmann, Los Altos, CA.

APT, R. K., AND PUGIN, J.-M. (1987), Maintenance of stratified databases viewed as a belief revision system, *in*, "Proceedings, ACM SIGACT-SIGMOD Symposium on Principles of Database System."

BIDOIT, N., AND FROIDEVAUX, C. (1987a), Minimalism subsumes default logic and circumscription in stratified logic programming, *in* "Proceedings, Logic in Computer Science (LICS-87), IEEE, New York," pp. 89–97.

BIDOIT, N., AND FROIDEVAUX, C. (1987b), "Minimalism Subsumes Default Logic and Circumscription in Stratified Logic Programming," Internal Report LRI No. 350.

BIDOIT, N., AND FROIDEVAUX, C. (1988a), More on stratified theories, *in* "Proceedings, European Conf. on Artificial Intelligence, Munich, August 1–5," pp. 492–494.

BIDOIT, N., AND FROIDEVAUX, C. (1988b), "Negation by Default and Non-stratifiable Logic Programs," Internal Report No. 437. To appear in a special issue of TCS on Research in Deductive Databases.

BIDOIT, N., AND HULL, R. (1986), Positivism vs. minimalism in deductive databases, *in* "Proceedings, ACM SIGACT-SIGMOD Symposium on Principles of Database System," pp. 123–132.

BIDOIT, N., AND HULL, R. (1989), Minimalism, justification and non-monotonicity in deductive databases, *J. Comput. System Sci.* **38**, No. 2, 290--325; L.R.I. Research Report 304, 1986.

BOSSU, G., AND SIEGEL, P. (1985), Saturation, nonmonotonic reasoning and the closed world assumption, *Artif. Intell.* **25**, 13–63.

CLARK, K. L. (1978), Negation as failure, *in* "Logic and Databases" (H. Gallaire and J. Minker, Eds.), pp. 293–322, Plenum, New York.

CHANDRA, A. K., AND HAREL, D. (1985), Horn clause queries and generalizations, *J. Logic Programming* **2**, No. 1, 1–15.

CLARK, K. L., AND TARNLUND, S. A. (Eds.), (1982), "Logic Programming," Academic Press, New York.

ETHERINGTON, D. W., AND REITER, R. (1983), On inheritance hierarchies with exceptions, *in* "Proceedings, AAAI-83," pp. 104–108.

FROIDEVAUX, C. (1986), Taxonomic default theory, *in* "Proceedings, European Conference on Artificial Intelligence, Brighton." pp. 123–129.

GELFOND, M. (1987), On stratified autoepistemic theories, *in* "Proceedings, AAAI-87, Seattle," pp. 207–211.

GELFOND, M., AND LIFSCHITZ, V. (1988), The stable model semantics for logic programming, *in* "Proceedings, Logic Programming Conf., Seattle," pp. 1070–1080.

GABBAY, D. M., AND SERGOT, H. J. (1986), Negation as inconsistency *I**, *J. Logic Programming* **1**, 1–35.

IMIELINSKI, T. (1985), Results on translating defaults to circumscription, *in* "Proceedings, 9th Int. Joint Conf. on Artificial Intelligence, Los Angeles," pp. 114–120.

KONOLIGE, K. (1987), On the relationship between default theories and autoepistemic logic, *in* "Proceedings, 10th Int. Joint Conf. on Artificial Intelligence, Milano," pp. 394–401.

LIFSCHITZ, V. (1988), On the declarative semantics of logic programs with negation, *in* "Foundations of Deductive Databases and Logic Programming (J. Minker, Ed.), pp. 177–192, Morgan Kaufmann, Los Altos, CA.

LLOYD, J. W. (1987), "Foundations of Logic Programming," 2nd ed., Springer-Verlag, New York.

LUKASZEWICZ, W. (1985), Two results on default logic, *in* "Proceedings, 9th Int. Joint Conf. on Artificial Intelligence, Los Angeles," pp. 459–461.

LUKASZEWICZ, W. (1988), Considerations on default logic: An alternative approach, *Comput. Intell.* **4**, 1–16.

MOORE, R. C. (1985), Semantical considerations on non-monotonic logic, *Artif. Intell.* **25**, 75–94.

MCCARTHY, J. (1980), Circumscription—A form of nonmonotonic reasoning, *Artif. Intell.* **13**, No. 1, 27–39.

MCDERMOTT, D., AND DOYLE, J. (1980), Nonmonotonic logic I, *Artif. Intell.* **13**, No. 1, 41–72.

NAQVI, S. A. (1986), A logic for negation in database systems, *in* "Workshop on Foundations of Deductive Databases and Logic Programming," pp. 378–387.

PRZYMUSINSKI, T. C. (1988), On the declarative semantics of stratified deductive databases, *in* "Foundations of Deductive Databases and Logic Programming" (J. Minker, Ed.), pp. 1193–216, Morgan Kaufmann, Los Altos, CA.

PRZYMUSINSKI, T. C., (1989), On the declarative and precedural semantics of logic programs, *J. Automat. Reason.* **5**, 167–205.

PRZYMUSINSKA, H., AND PRZYMUSINSKI, T. C. (1988), Weakly perfect model semantics for logic programs, *in* "Proceedings, Logic Programming Conf., Seattle," pp. 1106–1120.

REITER, R. (1980), A logic for default reasoning, *Artif. Intell.* **13**, No. 1, 81–132.

REITER, R., AND CRISCUOLO, G. (1981), On interacting defaults, *in* "Proceedings, Int. Joint Conf. on Artificial Intelligence, Vancouver," pp. 270–276.

VAN GELDER, A. (1988), Negation as failure using tight derivations for general logic programs, *in* "Foundations of Deductive Databases and Logic Programming" (J. Minker, Ed.), pp. 149–176, Morgan Kaufmann, Los Altos, CA.

VAN GELDER, A., ROTH, K., AND SCHLIPF, J. S. (1988), Unfounded sets and well founded semantics for general logic programs, *in* "Proceedings ACM SIGACT-SIGMOD Symp. on Principles of Database System," pp. 221–230.