Consistency of Clark's Completion and Existence of Stable Models

François Fages
LCR Thomson-CSF
and
LIENS, CNRS,
Ecole Normale Supérieure
45 rue d'Ulm, 75005 Paris
France
fages@dmi.ens.fr

1 Abstract

The most general notion of canonical model for a logic program with negation is the one of stable model [9]. In [7] the stable models of a logic program are characterized by the *well-supported* Herbrand models of the program, and a new fixed point semantics that formalizes the bottom-up truth maintenance procedure of [4] is based on that characterization. Here we focus our attention on the abstract notion of well-supportedness in order to derive sufficient conditions for the existence of stable models. We show that if a logic program Π is *positive-order-consistent* (i.e. there is no infinite decreasing chain w.r.t. the positive dependencies in the atom dependency graph of Π) then the Herbrand models of $comp(\Pi)$ coincide with the stable models of Π. From this result and the ones of [10] [17] [2] on the consistency of Clark's completion, we obtain sufficient conditions for the existence of stable models for positive-order-consistent programs. Then we show that a negative cycle free program can have no stable model if it is not positive-order-consistent, but that *order-consistency* alone [17] [2] (this condition generalizes call-consistency [10] [18], it is independent of positive-order-consistency) is sufficient to ensure the existence of a stable model. The last result is based on a characterization of stable models due to [5] and on a generalization of the result of [17] on the consistency of Clark's completion to infinite logic programs.

2 Introduction and Notations

A logic program Π is a finite set of rules of the form $L_1, ..., L_n \rightarrow A$, where A is an atom, called the *conclusion* and denoted by $concl(R)$, and the L_i's are literals (i.e. atoms or negated atoms), called the *premises* and denoted by $prem(R)$. We denote the subset of positive atoms in $prem(R)$ by $pos(R)$, and

the set of atoms under a negation by $neg(R)$. The set of ground instances of a logic program Π is denoted by $Ground(\Pi)$. Its Clark's completion [11] is denoted by $comp(\Pi)$.

The most general notion of canonical model for a logic program or its Clark's completion, is the one of stable model [9]. A Herbrand interpretation I of a logic program Π is said to be a *stable model* of Π if $I = M_{H(\Pi,I)}$, where $M_{H(\Pi,I)}$ denotes the least Herbrand model of the pure Horn program $H(\Pi, I)$ defined by the stability transformation:

$$H(\Pi, I) = \{pos(R) \rightarrow concl(R) \mid R \in Ground(\Pi) \ \wedge \ neg(R) \cap I = \emptyset\}$$

This two step transformation eliminates the rules which have inconsistent negative premises w.r.t. I, and eliminates the negative premises in the remaining rules. A model is stable if it derives itself by this transformation. The stable models of Π coincide with the default models of Π in Reiter's default theory [16] [13]. Any stable model of a logic program Π is a minimal Herbrand model of Π and of $comp(\Pi)$. Furthermore the stable model semantics coincides with various canonical model semantics defined for restricted classes of logic programs [9] [22] [12].

The aim of this paper is to investigate sufficient conditions for the existence of stable models. The central notion is the dependency graph of a logic program on which are based recent results on the completeness of SLDNF-resolution [10] and on the consistency of Clark's completion [17] [2].

The *predicate dependency graph* [1] of a logic program Π is a directed graph with signed edges. The nodes are predicate symbols occurring in Π. There is a positive (resp. negative) edge from p to q if there is a rule in Π with p in its positive premises (resp. negative premises) and q in the conclusion.

We say q *depends evenly (resp. oddly) on* p, denoted $p \leq_+ q$ (resp. $p \leq_- q$), if there is a path in the predicate dependency graph from p to q with an even (resp. odd) number of negative edges. q *depends on* p, denoted $p \leq q$, if $p \leq_+ q$ or $p \leq_- q$. We say q *depends positively on* p, denoted $p \leq_0 q$, if there is a non-empty path from p to q with all edges positive (this definition differs from [2]).

A logic program Π is *stratified* [1] [21] if no node depends on itself through at least one negative edge. Π is *call-consistent* [10] [18] if no node depends

oddly on itself, i.e. $\leq_-$ is irreflexive. For example the logic program $\{\neg a \rightarrow b, \quad \neg b \rightarrow a\}$ is call-consistent but not stratified.

2.1. Theorem [9] [22]. If Π is stratified then the stratified model of Π (see [1]) is the unique stable model of Π.

2.2. Theorem [18] [10] [2]. If Π is call-consistent then $comp(\Pi)$ has a Herbrand model.

The *atom dependency graph of* Π [15], denoted by $G(\Pi)$, is analogous to the predicate dependency graph. The nodes are the ground atoms of the Herbrand universe. There is a positive (resp. negative) edge from A to B if there is a rule $R \in Ground(\Pi)$ with $A \in pos(R)$ (resp. $A \in neg(R)$) and $B = concl(R)$. By abuse of notation we define also the relations $\leq$, $\leq_0$, $\leq_+$ and $\leq_-$, on ground atoms, in the same way as in the predicate dependency graph *.

A logic program Π is said to be *locally stratified* [15] if the relation of dependency through at least one negative edge in $G(\Pi)$ is well-founded. Π is said to be *negative cycle free* [17] if $\leq_-$ is irreflexive in $G(\Pi)$. Π is said to be *order-consistent* [17] if the relation ($\leq_+$ and $\leq_-$) in $G(\Pi)$ is well-founded (this condition is called local call-consistency in [2]). We say that Π is *positive-order-consistent* if $\leq_0$ is well-founded.

For example the logic program $\{p(s(X)) \rightarrow p(X), \quad \neg p(s(X)) \rightarrow p(X)\}$ is negative cycle free, but not order-consistent, nor positive-order-consistent because of the first rule, nor locally stratified because of the second rule. Note that on one hand order-consistency and positive-order-consistency are independent conditions, and on the other hand both classes of call-consistent programs and of locally stratified programs are contained in the class of order-consistent programs, itself contained in the class of negative cycle free programs.

2.3. Theorem [9] [22]. If Π is locally stratified then the unique perfect model of Π (see [15]) is the unique stable model of Π.

2.4. Theorem [17] [2]. If Π is order-consistent then $comp(\Pi)$ has a Herbrand model.

---

* Consistently with [17] we shall say that a binary relation (not necessarily a partial order) $\leq$ is well-founded if there is no infinite decreasing chain $x_0 \geq x_1 \geq \ldots$, in particular $\leq$ must be acyclic to be well-founded.

2.5. Theorem [17]. If Π is negative cycle free and either function free or internal variable free (i.e. for any rule the variables in the premise appear in the conclusion) then $comp(\Pi)$ has a Herbrand model.

The assumptions in theorems 2.4 and 2.5 are independent. It is an open problem whether negative cycle freeness alone implies the consistency of $comp(\Pi)$ or not. In the sequel we study sufficient conditions for the existence of stable models that subsume (local) stratification.

3 Existence of Stable Models, Part 1

The stable models of a logic program can be characterized in terms of well-supported interpretations. We say a Herbrand interpretation I is *well-supported* iff there exists a strict well-founded partial order $\prec$ on I such that for any atom $A \in I$ there exists a rule $R \in Ground(\Pi)$ with $concl(R) = A$, $I \models prem(R)$ and for any $B \in pos(R)$, $B \prec A$.

3.1. Theorem [7]. For a general logic program Π, the well-supported models of Π are exactly the stable models of Π.

This was shown independently by [6] in the case where Π is a propositional program. For example the program $\Pi_2 = \{p \rightarrow p,\ \neg p \rightarrow q\}$ has two supported minimal models, $\{p\}$ and $\{q\}$ which are both models of $comp(\Pi_2)$, but only one well-supported model $\{q\}$ which is also the unique stable model and the iterated least model in the stratified semantics of [1] [21]. The condition of well-supportedness eliminates cyclic and infinite supports. Well-supported models are finitely justified models. For instance the program $\Pi_3 = \{p(s(x)) \rightarrow p(x)\}$, given with a constant a, has two supported Herbrand models, $\emptyset$ and $\{p(s^i(a)) | i \geq 0\}$. They are both models of $comp(\Pi_3)$, but $\emptyset$ is the only finitely justified model of Π_3, i.e. the unique stable model of Π_3.

3.2. Theorem. If Π is positive-order-consistent then the Herbrand models of $comp(\Pi)$ coincide with the stable models of Π.

Proof. A stable model of Π is a Herbrand well-supported model of Π, so it is a Herbrand minimal supported model of Π, hence a model of $comp(\Pi)$ [11]. Conversely let us reason by contradiction. Suppose M is a Herbrand model of $comp(\Pi)$ but not a well-supported model of Π. There exists an atom $A \in M$ that cannot be finitely justified. However as M is a supported model of Π, there exists a rule $R \in Ground(\Pi)$ with $concl(R) = A$ and $M \models prem(R)$. As

A cannot be finitely justified there exists $B \in pos(R)$, so $B \leq_0 A$, such that B cannot be finitely justified. Therefore we get an infinite decreasing chain w.r.t. $\leq_0$, i.e. a contradiction with the positive-order-consistency of Π. □

Since stable models are minimal Herbrand models [9] we obtain:

3.3. Corollary. If Π is positive-order-consistent then all Herbrand models of $comp(\Pi)$ are minimal.

3.4. Corollary. If Π is positive-order-consistent, negative cycle free and either function free or internal variable free, then Π has a stable model.

Proof. By 3.2 and 2.5. □

3.5. Corollary. If Π is positive-order-consistent and order-consistent then Π has a stable model.

Proof. By 3.2 and 2.4. □

3.6. Corollary. If Π is call-consistent and the relation $\leq_0$ on predicate symbols is acyclic (these assumptions are clearly decidable) then Π has a stable model.

Proof. By 3.5, as call-consistency implies order-consistency and $\leq_0$ acyclic on predicate symbols implies positive-order-consistency. □

These corollaries can be exploited by logic programming tools based on the stable model semantics, as in Truth Maintenance Systems [7] [6] or in Deductive Data Bases [20] [23], to detect semantic errors statically. They are also interesting from a programming point of view. Any logic program can be transformed into a positive-order-consistent program that preserves the program's completion semantics. The transformation consists in replacing a positive premise in a rule

$$L_1, ..., p(t_1, ..., t_k), ..., L_n \rightarrow A$$

by a negative premise

$$L_1, ..., \neg\overline{p}(t_1, ..., t_k), ..., L_n \rightarrow A$$

adding the rule

$$\neg p(x_1, ..., x_k) \rightarrow \overline{p}(x_1, ..., x_k)$$

It is obvious that this double negative does not change the program's completion semantics, but it makes $\leq_0$ acyclic on predicate symbols. That transformation appears in [19] to show that the program's completion semantics and the well-founded semantics have the same theoretical expressive power. From a logic programming point of view under the stable model semantics that transformation is a way to relax the constraint of well-supportedness for some predicates, by requiring only that the intended interpretation of these predicates be supported, as in the models of the program's completion.

It is an open problem to know whether negative cycle free programs in the general case have a consistent completion. However we show with an example that this condition does not ensure the existence of a stable model.

3.7. Theorem. There exist (internal variable free) negative cycle free programs that have no stable model.

Proof. Let $\Pi = \{p(s(X)) \rightarrow p(X), \quad \neg p(s(X)) \rightarrow p(X)\}$. This is an internal variable free negative cycle free program, so $comp(\Pi)$ is consistent by 2.5. The only Herbrand model of $comp(\Pi)$ takes p true everywhere, however this is not a well-supported model as it is not finitely justified. Therefore Π has no stable model. □

So positive-order-consistency cannot be eliminated in the assumption of corollary 3.4. However positive-order-consistency (resp. $\leq_0$ acyclic) is not necessary in the assumption of corollary 3.5 (resp. 3.6). To show this we study first the Clark's completion of infinite logic programs.

4 Consistency of Completed Infinite Logic Programs

An *infinite logic program* Π is an infinite set of (finite) rules

$$L_1, ..., L_n \rightarrow A$$

formed on a countable set of variables and a finite alphabet of constants, functions and predicates. In this way $Ground(\Pi)$ is always countable. The atom dependency graph is defined as for finite programs. In this section we show that

theorem 2.4 holds for infinite logic programs as well, with the same proof as in [17] [2] in the line of [10] [18].

The Clark's completion of an infinite logic program is naturally defined by an infinite formula of first-order classical logic. The *completed definition* of a predicate p, supposed to be unary here, is an infinite first-order formula of the form

$$\forall x\ p(x) \leftrightarrow \exists y_1 ... \exists y_j ...\ (x = t_1 \wedge C_1) \vee ... \vee\ \ (x = t_i \wedge C_i) \vee ...$$

where the disjunction can be infinite, but each member $(x = t_i \wedge C_i)$ is a finite conjunction corresponding to a rule $R \in \Pi$ with $concl(R) = p(t_i)$ and $prem(R) = C_i$. The *Clark's completion* of an infinite logic program is composed of the (finite) conjunction of the completed definition of each predicate symbol, together with the strong equality axioms.

The immediate consequence operator is defined undifferently on infinite logic programs,

$$T_\Pi(I) = \{concl(R) \mid R \in Ground(\Pi),\ I \models prem(R)\}$$

Its fixed points are the Herbrand models of $comp(\Pi)$.

4.1. Proposition. Let Π be an infinite logic program. Then a Herbrand interpretation I is a model of $comp(\Pi)$ if and only if I is a fixed point of T_Π.

Proof. One can easily check that the proof in [11] for definite programs remains correct in presence of both negation [1] and infinite completed definitions. □

Now, pair mappings on the set of *partial interpretations* in $2^{B_H} \times 2^{B_H}$, first introduced in [8], can be associated to infinite programs without any modification. Let Π be an infinite logic program and $(M, N) \in 2^{B_H} \times 2^{B_H}$ be a partial interpretation, let us define

$$T^+_\Pi(M,N) = \{concl(R) \mid R \in Ground(\Pi),\ pos(R) \subseteq M,\ neg(R) \subseteq N\}$$

$$T^-_\Pi(M,N) = \{A \mid \forall R \in Ground(\Pi)\ \ concl(R) = A\ \Rightarrow\ pos(R) \cap N \neq \emptyset \vee neg(R) \cap M \neq \emptyset\}$$

The pair mapping $< T^+_\Pi, T^-_\Pi >$ on infinite logic programs enjoys the same property of monotonicity and gives Herbrand models of $comp(\Pi)$ under certain conditions.

4.2. Proposition. If $M \cap N = \emptyset$ then $T^+_\Pi(M,N) \cap T^-_\Pi(M,N) = \emptyset$.

4.3. Proposition. $< T^+_\Pi, T^-_\Pi >$ is monotonic in the lattice $2^{B_H} \times 2^{B_H}$ ordered by pair inclusion $\sqsubseteq$, that is $(M_1, N_1) \sqsubseteq (M_2, N_2)$ (i.e. $M_1 \subseteq M_2$ and $N_1 \subseteq N_2$) implies $< T^+_\Pi, T^-_\Pi > (M1, N1) \sqsubseteq < T^+_\Pi, T^-_\Pi > (M2, N2)$.

4.4. Proposition. If $M \cap N = \emptyset$ and $(M,N) \sqsubseteq < T^+_\Pi, T^-_\Pi > (M,N)$ then there exists a fixed point (M', N') of $< T^+_\Pi, T^-_\Pi >$ such that $(M,N) \sqsubseteq (M', N')$ and $M' \cap N' = \emptyset$.

4.5. Proposition. If (M,N) is a fixed point of $< T^+_\Pi, T^-_\Pi >$, $M \cap N = \emptyset$ and $M \cup N = B_H$ then M is a Herbrand model of $comp(\Pi)$.

Proof. As $M \cap N = \emptyset$ and $M \cup N = B_H$ we have $T^+_\Pi(M,N) = T_\Pi(M)$ by definition. As (M,N) is a fixed point of $< T^+_\Pi, T^-_\Pi >$ we have $T^+_\Pi(M,N) = M$. Therefore M is a fixed point of T_Π, i.e. a model of $comp(\Pi)$ by 4.1. □

At this point it is clear that the useful properties of pair mappings and order-consistency exploited in [17] on finite logic programs are met by infinite logic programs. Therefore one can check that through the same series of lemmas with the same proofs as in [17] we get:

4.6. Theorem. If an infinite logic program Π is order-consistent then $comp(\Pi)$ has a Herbrand model.

On the other hand, note that the proof of existence of a Herbrand model for completed negative cycle free programs in [17] does make use of the compactness theorem of first-order classical logic, hence it cannot be lifted to infinite logic programs. In fact there exist infinite negative cycle free programs, like

$$\{\neg P(s^i(x)) \rightarrow P(x) \mid i > 0\},$$

which have an inconsistent Clark's completion.

5 Existence of Stable Models, Part 2

This section makes use of the characterization due to [5] of the stable models of a logic program Π by the Herbrand models of its *fixpoint completion*. We recall here the basic definitions with our notations.

A *quasi-interpretation* Q is a possibly infinite set of ground rules without positive premise, i.e. of the form

$$\neg A_1, ..., \neg A_n \rightarrow A$$

where $n \geq 0$, and A and the A_i's are ground atoms formed over a finite alphabet.

Given a (finite) logic program Π, the *immediate consequence operator* T_Π is defined on quasi-interpretations by

$$\begin{aligned} T_\Pi(Q) = \{ & neg(R_1), ..., neg(R_n), neg(R) \rightarrow A \ \mid \ R \in Ground(\Pi) \\ & concl(R) = A,\ pos(R) = \{A_1, ..., A_n\},\ n \geq 1, \\ & \forall i,\ 1 \leq i \leq n,\ R_i \in Q,\ concl(R_i) = A_i \} \end{aligned}$$

T_Π is a continuous operator in the lattice of quasi-interpretations [5]. The least fixed point of T_Π is denoted by $fix(\Pi)$, i.e.

$$fix(\Pi) = \bigcup_{i \geq 0} T_\Pi \uparrow i$$

In general $fix(\Pi)$ is an infinite quasi-interpretation, its Clark's completion is then defined as in the previous section. We shall use the equivalence theorem of [5] that relates the stable models of Π with the Herbrand models of $comp(fix(\Pi))$.

5.1. Theorem [5]. The Herbrand models of $comp(fix(\Pi))$ are exactly the stable models of Π.

5.2. Lemma. Let Π and Π' be two (infinite) logic programs based on the same alphabet. The relation $\leq_+$ (resp. $\leq_-$) in $G(\Pi)$ is included in $\leq_+$ (resp. $\leq_-$) in $G(\Pi')$ if and only if for each rule $R \in Ground(\Pi)$ with $concl(R) = B$, for every $A \in pos(R)$ (resp. $A \in neg(R)$), we have $A \leq_+ B$ (resp. $A \leq_- B$) in $G(\Pi')$.

Proof. If $\leq_+$ (resp. $\leq_-$) in $G(\Pi)$ is included in $\leq_+$ (resp. $\leq_-$) in $G(\Pi')$ then for any rule $R \in Ground(\Pi)$ with $concl(R) = B$, and for every $A \in pos(R)$ (resp. $A \in neg(R)$) we have $A \leq_+ B$ (resp. $A \leq_- B$) in $G(\Pi)$. Thus by hypothesis we get $A \leq_+ B$ (resp. $A \leq_- B$) in $G(\Pi')$.

Conversely if $A \leq B$ in $G(\Pi)$ then there is a path from A to B in $G(\Pi)$. For each positive (resp. negative) edge (A_i, A_{i+1}) in that path there exists a rule $R \in Ground(\Pi)$ with $concl(R) = A_{i+1}$ and $A_i \in pos(R)$ (resp. $A_i \in neg(R)$). So by hypothesis we have $A_i \leq_+ A_{i+1}$ (resp. $A_i \leq_- A_{i+1}$) in $G(\Pi')$. Therefore if $A \leq_+ B$ (resp. $A \leq_- B$) in $G(\Pi)$, there is an even (resp. odd) number of negative edges in the path, thus we get $A \leq_+ B$ (resp. $A \leq_- B$) in $G(\Pi')$. $\square$

5.3. Lemma. If $A \leq_- B$ in $G(fix(\Pi))$ then $A \leq_- B$ in $G(\Pi)$.

Proof. By 5.2 it suffices to show that for any rule $R \in fix(\Pi)$, i.e. for any $i \geq 0$ and any $R \in T_\Pi \uparrow i$, we have $A \leq_- B$ in $G(\Pi)$ if $concl(R) = B$ and $A \in neg(R)$ (the case $A \in pos(R)$ does not arise as $fix(\Pi)$ is a quasi-interpretation). The proof is by induction on i.

The base case is trivial. Let $R \in T_\Pi \uparrow i+1$, let $A \in neg(R)$ and $concl(R) = B$. By definition of $T_\Pi \uparrow i+1$ there exists a rule $S \in Ground(\Pi)$ with $concl(S) = B$ and either $A \in neg(S)$ or $A \in neg(R')$ with $R' \in T_\Pi \uparrow i$ and $concl(R') \in pos(S)$. In the former case we get immediately $A \leq_- B$ in $G(\Pi)$. In the latter case we get $A \leq_- concl(R')$ in $G(\Pi)$ by induction. As $concl(R') \leq_+ B$ in $G(\Pi)$ we conclude again $A \leq_- B$ in $G(\Pi)$. □

5.4. Theorem. An order-consistent logic program has a stable model.

Proof. If a logic program Π is order-consistent then as $fix(\Pi)$ is a quasi-interpretation, $\leq_0$ is the identity relation, so we get by lemma 5.3 that $fix(\Pi)$ is also order-consistent. Therefore by theorem 4.6 $comp(fix(\Pi))$ has a Herbrand model, hence Π has a stable model by theorem 5.1 . □

6 Conclusion

We have shown that the main syntactic criteria that ensure the consistency of Clark's completion of logic programs [10] [18] [17] [2], ensure also the existence of stable models for these programs, with the noticeable exception of negative cycle free programs that can have no stable model. The most general conditions (order-consistency on one hand, negative cycle freeness and positive-order-consistency on the other hand) are probably undecidable, but they have a checkable counterpart, namely call-consistency. This can be directly exploited by logic programming tools based on the stable model semantics, as in Truth Maintenance Systems [7] [6] or in Deductive Data Bases [20] [23], to detect semantic errors statically.

One remaining problem is to find a direct proof of the existence of stable models for order-consistent logic programs. For example every stable model of a logic program can be obtained as an ordinal power of the justification maintenance operator associated to the program under a suitable strategy [7]. One would like to exhibit for order-consistent programs a class of strategies under

which the ordinal powers of the justification maintenance operator are increasing, whence reach a stable model. In [7] it is shown that any fair strategy constructs the 2-valued well-founded model of the program if it exists. Here the proof of existence of a stable model under the order-consistency assumption suggests another transfinite process: first construct the fixpoint completion by iterating T_Π up to ordinal ω, then choose a signing, that gives a stable model, for example by iterating the justification maintenance operator under any conservative strategy (i.e. any strategy that selects in priority the rules which do not have for effect to retract an atom). A fundamental problem underlying these difficulties is the logical complexity of stable models for restricted classes of programs [19].

The existence of stable models for order-consistent (or call-consistent) logic programs can be transcripted in various theories of non-monotonic reasoning developed in AI. In Truth Maintenance System terminology, we have shown that if a set of justifications contains no odd loop in its dependency network then there exists a state of beliefs with well-founded supporting justifications, or equivalently (see [7]) there exists a strategy under which the Truth Maintenance procedure converges. In Reiter's default logic we get that an order-consistent default theory is consistent. In Moore 's autoepistemic logic, we get the existence of a stable autoepistemic expansion for any call-consistent set of premises.

Finally we conjecture that although negative cycle free programs can have no stable model, their Clark's completion is always consistent. In case of a positive answer to the conjecture, negative cycle freeness would subsume the other conditions that ensure the consistency of the Clark's completion, but with no guarantee on the existence of canonical model for the program.

References

[1] K.R. Apt, H.A. Blair, A. Walker, *Towards a theory of declarative knowledge*, in Foundations of deductive databases and logic programming, Minker, J. (ed.), Morgan Kaufmann, Los Altos (1987).

[2] A. Cortesi, G. Filé, *Graph properties for normal logic programs*, Proc. 5th Conv. Nat. sulla Programmazione Logica GULP'90, Padova, (1990).

[3] L. Cavedon, J.W. Lloyd, *A completeness theorem for SLDNF-resolution*, Journal of Logic Programming, 7(3), pp.177-192 (1989).

[4] J. Doyle, *A truth maintenance system*, Artificial Intelligence, vol. 12, pp.231-272 (1979).

[5] P.M. Dung, K. Kanchanasut, *A fixpoint approach to declarative semantics of logic programs*, NACLP'89. MIT Press (1989).

[6] C. Elkan, *A rational reconstruction of nonmonotonic truth maintenance systems*, Journal of Artificial Intelligence, vol. 43, pp.219-234 (1990).

[7] F. Fages, *A new fixpoint semantics for general logic programs compared with the well-founded and the stable model semantics*, 7th International Conference on Logic Programming, Jerusalem. MIT Press (June 1990). To appear in the Journal of New Generation Computing.

[8] M.R. Fitting, *A Kripke-Kleene semantics for logic programs*, J. of Logic Programming 2, pp.295-312 (1985).

[9] M. Gelfond, V. Lifschitz, *The stable model semantics for logic programming*, Proc. of the 5th Logic Programming Symposium, pp.1070-1080, MIT press (1988).

[10] K. Kunen, *Signed data dependencies in logic programs*, Journal of Logic Programming, 7(3), pp.231-245 (1989).

[11] J.W. Lloyd, *Foundations of Logic Programming*, Springer Verlag (1987).

[12] W. Marek, V.S. Subrahmanian, *The relationship between logic program semantics and non-monotonic reasoning*, Proc. Sixth International Conference on Logic Programming, Lisbon (1989).

[13] W. Marek, M. Truszcinski, *Stable semantics for logic programs and default theories*, NACLP'89. MIT Press (1989).

[14] T. Przymusinski, *On the declarative and procedural semantics of logic programs*, Journal of Automated Reasoning, 5, pp.167-205 (1989).

[15] T. Przymusinski, *On the declarative semantics of stratified deductive data-bases and logic programming*, in Foundations of deductive databases and logic programming, Minker, J. (ed.), Morgan Kaufmann, Los Altos (1987).

[16] R. Reiter, *A logic for default reasoning*, Artificial Intelligence, 13, pp.81-132 (1980).

[17] T. Sato, *Completed logic programs and their consistency*, Journal of Logic Programming, vol.9 (1), pp.33-44 (1990).

[18] T. Sato, *On the consistency of first-order logic programs*, ETL technical report TR-87-12, (1987).

[19] J.S. Schlipf, *The expressive powers of the logic programming semantics*, PODS'90, pp.196-204, (1990).

[20] D. Sacca, C. Zaniolo, *Stable models and non-determinism in logic programs with negation*, PODS'90, pp.205-217, (1990).

[21] A. Van Gelder, *Negation as failure using tight derivations for general logic programs*, in Foundations of deductive databases and logic programming, Minker, J. (ed.), Morgan Kaufmann, Los Altos (1987). Also in J. of Logic Programming 1989 pp.109-133.

[22] A. Van Gelder, K. Ross, J.S. Schlipf, *Unfounded sets and well-founded semantics for general logic programs*, Proc. of the Symp. on Principles of Databases Systems, ACM-SIGACT-SIGCOM (1988).

[23] D.S. Warren, *The XWAM: a machine that integrates Prolog and deductive database query evaluation*, Technical Report 89/25, Suny at Stony Brook, NY (1989).