Semantics for Reasoning with Contradictory Extended Logic Programs

Anastasia Analyti & Sakti Pramanik
Department of Computer Science, Michigan State University, E. Lansing, MI 48824
E-mail: analyti@cps.msu.edu, pramanik@cpswh.msu.edu

Abstract

Extended programs are normal programs extended with classical negation. Because of the classical negation in the head of the rules, an extended logic program can be contradictory. Human reasoning is often based on conflicting evidence and on assumptions which are not always valid. Our goal is to derive useful conclusions from programs that may be contradictory. We consider rules to be defaults. Rule prioritization can be viewed as a tool to specify confidence information about these defaults. We present a new semantics for extended programs with rule prioritization, called *contradiction-free semantics* (*CFS*). *CFS* is defined as the least fixpoint of a monotonic operator. Every extended program with rule prioritization has at least one *stable c-model*. We show that the *CFS* of a program *P* coincides with the least stable c-model of *P*. A sound and complete proof procedure to answer queries based on *CFS* is described.

Keywords: extended logic programs, rule prioritization, contradictions, declarative semantics, procedural semantics.

1. Introduction

Extended programs provide negative information both implicitly (negation by default ~) and explicitly (classical negation ¬). Classical negation is needed: (i) in case of incomplete information, since it may not be justified for a particular information to be considered false because of absence of further information (closed world reasoning), (ii) when negative information should be inferred if some conditions are satisfied, for example, ¬light_off ← light_on, and (iii) to represent default reasoning and exceptions, for example, some of the exceptions of the general rule fly(X)← bird(X) are ¬fly(X)← ostrich(X), ¬fly(X)← penguin(X).

Several semantics for extended programs have been proposed in the literature [Przy90, GeLi91, DuRu91, PAA91*a*, PAA92, PeAl92, Witt92, Dung93, Wagn93]. Yet, these semantics are not defined for all extended programs. In [Przy90], the *well-founded model* [vGRS91] of an extended program *P* is computed as that of a normal program after replacing every literal ¬*L* of *P* with a new atom ¬_*L*. However, the well-founded model of an extended program can be *contradictory*. For example, the well-founded model of P={$\neg p \leftarrow \sim a.\ \ p\leftarrow.\ \ b\leftarrow.$} is {$\sim a, \neg p, p, b$} and because of the contradiction, *P* is not given any semantics in [Przy90]. However, intuitively, the rule $b\leftarrow$ is not "suspect" for the derivation of literals p, $\neg p$ in *P* and thus *b* should be true.

The *contradiction removal semantics* (*CRS*), defined in [PAA91*a*, PAA92], extends the *well-founded semantics* [vGRS91] and avoids contradictions brought about by closed-world assumptions (*CWAs*). For example, the *CRS* of P={$\neg p \leftarrow \sim a.\ \ p\leftarrow.\ \ b\leftarrow.$} is {$p, b$} which is non-contradictory. Yet, the problem is not totally solved since no semantics is given to P'={$\neg p\leftarrow.$

$p \leftarrow .\ b \leftarrow .\}$ even though b should be true. The same arguments hold for the *argumentation semantics* [Dung93].

Human reasoning is often based on conflicting evidence and on assumptions which are not always valid. Our goal is to derive useful conclusions from programs that may be contradictory. We consider rules to be defaults. Rule prioritization can be viewed as a tool to specify confidence information about these defaults. Rule prioritization is investigated in [LaVe90, GLV91, GeVe91, BrLi91, BrLi93, LaVe92, LeRu92]. Yet, negation by default is not considered in these works. In [LaVe90, GLV91, GeVe91, LaVe92, LeRu92], alternative semantics for *ordered logic* programs are presented. A default in an ordered logic program is a unidirectional rule. In [BrLi91, BrLi93], a default is a clause, that is, there is no distinction between the head and the body of a default rule. A conceptualization of both implicit and explicit preferences on data is given in [Hunt92].

A *prioritized extended program* (*PEP*) consists of a set of partially ordered rules. Every rule r has a corresponding set $C_r \subseteq Body_r$[1] which is called the *contrapositive set* of r. Intuitively, an extended program P is contradictory when there is a literal L such that both L and $\neg L$ are derived in P. In this case, there is a rule r such that both $Body_r$ and $\neg Head_r$ are derived from P. The value of C_r indicates which rules are "suspect" for the contradiction. When $C_r=\{\}$, only the rule r is "suspect". When $C_r \neq \{\}$, the rules used in the derivation of the literals in C_r are considered "suspect". To facilitate this reasoning, P is expanded with the contrapositives r' of every rule r such that $Head_{r'} \in \{\neg L \mid L \in C_r\}$. When contradiction occurs, rule prioritization indicates the relative reliability of the "suspect" rules. We define the *contradiction-free semantics* (*CFS*) of a *PEP*. *CFS* is always defined and non-contradictory. Every *PEP* has at least one *stable c-model*. The *CFS* of a program P is the least[2] fixpoint of a monotonic operator and the least grounded c-model of P. When the Herbrand base is finite, the complexity of computing *CFS* is polynomial w.r.t. the size of the program.

CFS extends the well-founded semantics for normal programs [vGRS91] to *PEPs*. The use of contrapositives for resolving contradictions in *CFS* has been supported by [GiMa90, Witt91]. Yet, in these works, rule prioritization is not considered and $C_r=Body_r$ for every rule r. Let P be an extended program with $C_r=Body_r$ for every rule r. Then, the *CFS* of P is a subset of the *generalized stable model semantics* of P [GiMa90], if the latter is defined. Our semantics is also related to ordered logic [GLV91, LaVe92]. An ordered logic program can be seen as a *PEP* which is free of default literals and $C_r=\{\}$ for every rule r. If P is an ordered logic program then the *CFS* of P coincides with the *skeptical c-partial model* of P [GLV91] and is a subset of the *well-founded partial model* of P [LaVe92].

The rest of the paper is organized as follows. In Section 2, we define the c-models of a *PEP*. In Section 3, we define the *CFS* and grounded c-models of a *PEP*. We show that the *CFS*

[1] $Body_r$ denotes the set of literals in the body of rule r and $Head_r$ denotes the head of rule r.

[2] A set I is the *least* element of a set $\mathbf{I}$ iff $I \in \mathbf{I}$ and $I \subseteq J$, for all $J \in \mathbf{I}$.

of a *PEP*, *P*, coincides with the least grounded c-model of *P*. In Section 4, we compare *CFS* with other semantics. In Section 5, we present the procedural semantics for *CFS*. Section 6 contains the concluding remarks.

2. C-models for Prioritized Extended Programs

Our alphabet contains a finite set of constant, predicate and variable symbols from which terms and atoms are constructed in the usual way. A *classical literal* is either an atom A or its classical negation $\neg A$. The classical negation of a literal L is denoted by $\neg L$ and $\neg(\neg L)=L$. The symbol ~ stands for negation by default and $\sim(\sim L)=L$, $\neg(\sim L)=L$. A *default literal* is denoted by $\sim L$, where L is a classical literal.

A *prioritized extended program* (*PEP*) is a tuple $P=<R_P,<_{\mathbf{R}}>$. R_P is a set of rules r: $L_0\leftarrow L_1,...,L_m,\sim L_{m+1},...,\sim L_n$, where r is a label and L_i are classical literals. Every rule r has a corresponding set $C_r \subseteq Body_r$, called the *contrapositive set* of r. The precise meaning of C_r will be given in the definitions. Intuitively, when there is a rule r such that both $Body_r$ and $\neg Head_r$ are derived from P, the value of C_r indicates the "suspects" for the contradiction. When C_r={}, the rule r is considered incomplete[3]. When $C_r \neq \{\}$, the contradiction is considered as evidence that one of the literals in C_r was wrongly derived. Thus, the *CWAs* and/or rules used in some step of the derivation of literals in C_r are considered unreliable. To facilitate this reasoning, P is expanded with the contrapositives r' of every rule r such that $Head_{r'}\in\{\neg L| L\in C_r\}$.

For example, consider the program $P=\{r_1$: a. r_2: b. r_3:$p\leftarrow a$. r_4: $\neg p\leftarrow b$. with $C_{r3}=\{\}$ and $C_{r4}=\{\}\}$. Because both a, $\neg p$ are derived in P and $C_{r3}=\{\}$, the rule r_3 is considered incomplete, i.e., r_3 should be $p\leftarrow a,\sim\neg p$. A similar argument applies to rule r_4. Thus, literals a, b can be reliably evaluated as true but the truth value of p is unknown. In contrast, consider the program $P'=\{r_1$: a. r_2: b. r_3:$p\leftarrow a$. r_4: $\neg p\leftarrow b$. with $C_{r3}=\{a\}$ and $C_{r4}=\{b\}\}$. Since $C_{r3}=\{a\}$, the rule r_1 used for the derivation of a is considered unreliable. Similarly, the rule r_2 used for the derivation of b is considered unreliable. By expanding P' with the contrapositives r'_3: $\neg a\leftarrow \neg p$ and r'_4: $\neg b\leftarrow p$, the derivation of a, b from rules r_1, r_2 is blocked and the literals a, b, p are evaluated as unknown. The view C_r={} for every rule r is implicit in ordered logic [GLV91, LaVe92] and vivid logic [Wagn93]. The view $C_r = Body_r$ for every rule r is adopted in [GiMa90, Witt91]. Yet, other views such as $C_r\neq\{\}$ and $C_r \neq Body_r$ for a rule r are also possible.

The relation $\lessdot_{\mathbf{R}} \subseteq R_P\times R_P$ is a strict partial order (irreflexive, asymmetric and transitive), denoting the relative reliability of the rules. Let r and r' be two rules. The notation $r\lessdot r'$ means that r is less reliable than r', that is, $r\lessdot r'$ iff $(r,r')\in <_{\mathbf{R}}$. The notation $r \not\lessdot r'$ means that r is not less

[3] We say that a rule is *incomplete* if not all possible exceptions are enumerated in its body.

reliable that r'. Note that, $r \not< r$ since $<_{\mathbf{R}}$ is irreflexive. Intuitively, when $Body_r$ is true, $Head_r$ is evaluated as true iff $\neg Head_r$ cannot be derived from rules with priority no lower than r. Thus, deciding if $Head_r$ is true depends only on the rules $r' \not< r$. Note that a *PEP* with $<_{\mathbf{R}}=\{\}$ is an extended logic program.

The set of instantiated classical literals of P is called the *Herbrand Base* (HB_P) of P. The instantiation of a *PEP*, P, is defined as follows: The instantiation of R_P is defined the usual way. Let r_{inst} and r'_{inst} be instances of rules r and r' in P then $r_{\mathrm{inst}}<r'_{\mathrm{inst}}$ iff $r<r'$. In all sections but Section 5, we assume that programs have been instantiated and thus all rules are propositional. If S is a set of literals then $\sim S =_{\mathrm{def}} \{\sim L \mid L \in S\}$ and $\neg S =_{\mathrm{def}} \{\neg L \mid L \in S\}$. If L is a literal then $\neg(\sim L)=L$.

Definition 2.1 (program expansion): Let P be a *PEP*. The *expansion* $exp(P)$ of P is also a *PEP*, defined as follows:

- For every rule r: $H \leftarrow L_1,...,L_n$ of P, $exp(P)$ contains the rules $\{r'_i: \neg L_i \leftarrow L_1,...,L_{i-1},L_{i+1},...,L_n,\neg H \mid L_i \in C_r$ and $C_{r'_i}=(C_r-\{L_i\})\cup\{\neg H\}\}$ (called *contrapositives* of r) and the rule r.
- The partial ordering of the rules of P is extended to the rules of $exp(P)$ as follows: If r and r' are two rules of P with $r<r'$ (resp. $r \not< r'$) then r and any contrapositive of r has less (resp. neither less nor more) priority than r' and any contrapositive of r'. If r and r' are contrapositives then $r \not< r'$.

Note that $exp(exp(P))=exp(P)$.

Definition 2.2 (interpretation): Let P be a *PEP*. A set $I=T\cup\sim F$ is an interpretation of P iff T and F are disjoint subsets of HB_P. An interpretation I is *consistent* iff there is no L such that both $L\in T$ and $\neg L\in T$. An interpretation I is *coherent* iff it satisfies the *coherence property*: $L\in F$ if $\neg L\in T$.

In interpretation $I=T\cup\sim F$, T contains the *classically true* literals, $\neg T$ contains the *classically false* literals and F contains the literals *false by default*. The *coherence property* first appeared in [PeAl92] and it expresses that if a literal is classically false then it is also false by default.

Definition 2.3 (truth valuation of a literal): A literal L is true (resp. false) w.r.t. an interpretation I iff $L\in I$ (resp. $\sim L\in I$). A literal that is neither true nor false w.r.t. I, it is undefined w.r.t. I.

An interpretation I can be seen equivalently as a function from the set of ground classical literals to $\{0,1/2,1\}$, where $I(L)=1$ when L is true w.r.t. I, $I(L)=0$ when L is false w.r.t. I and $I(L)=1/2$ when L is undefined w.r.t. I. Both views of an interpretation, as a set and as a function, will be used in the paper. Note that, $I(\sim L)=1-I(L)$, for any literal L. If I is a coherent interpretation then $I(L)=1$ implies $I(\neg L)=0$. We define $I(\emptyset)=_{\mathrm{def}} 1$ and $I(S)=_{\mathrm{def}} \min\{I(L) \mid L\in S\}$ where S is a non-empty set of literals. The *coherence operator* (*coh*) [PeAl92] transforms an interpretation to a coherent one. Let $I=T\cup\sim F$ be an interpretation of a *PEP*. Then, ***coh***(I) is the coherent interpretation $T\cup\sim F'$, where $F'=F\cup\{L \mid \neg L\in T\}$.

In Definition 2.4, the concept of unfounded classical literal w.r.t. a rule r and interpretation I is defined. I represents the set of literals known to be true. This concept is used in the fixpoint computation of CFS. In particular, a rule r is used for the derivation of $Head_r$ only if $\neg Head_r$ is unfounded w.r.t. r and CFS. Intuitively, a literal L is unfounded w.r.t. r and I if L cannot be derived from the rules $r' \not< r$ when literals are assumed to be false as indicated in I.

Definition 2.4 (unfounded literal w.r.t. *r* and *I*): Let P be a PEP, r a rule and I an interpretation. A classical literal L is called unfounded w.r.t. r and I iff there is a classical literal set S s.t. $L \in S$ and $\forall H \in S$: If r' is a rule in $exp(P)$ s.t. $Head_{r'} = H$ and $r' \not< r$ then (i) $I(Body_{r'})=0$ or (ii) there is a classical literal $L' \in C_{r'}$ s.t. $L' \in S$ or (iii) there is a default literal $\sim L' \in C_{r'}$ s.t. $\neg L' \in S$.

Note that if a literal L is unfounded w.r.t. r and I then L is unfounded w.r.t. r and any interpretation $I' \supseteq I$. If a rule r is unidirectional ($C_r=\{\}$) and $I(Body_r) \neq 0$ then $Head_r$ is not unfounded w.r.t. any rule r' s.t. $r \not< r'$. Intuitively, when $C_r = Body_r$ $\forall$rule r, every rule is given higher priority than the $CWAs$. In Example 2.1, we show that this is not true when there is a literal $L \in Body_r - C_r$ for a rule r. An algorithm that decides if a literal L is unfounded w.r.t. r and I is given in Appendix A. The time-complexity of the algorithm is linear w.r.t. the size of P.

Example 2.1: Let P be the expanded (with contrapositives) PEP:

$R_P = \{r_1: fly. \quad r_2: \neg fly \leftarrow \sim bird. \; r'_2: bird \leftarrow fly.$ with $C_{r_2} = \{\sim bird\}$, $C_{r'_2} = \{fly\}\}$ and $<_{\mathbf{R}} = \{\}$.

Then, the literal $\neg fly$ is unfounded w.r.t. r_1 and Ø (in Def. 2.4 take $S=\{\neg fly, \neg bird\}$). This implies that fly can be reliably derived from rule r_1. Intuitively, in this case, rule r_1 is given higher priority than the CWA, $\sim bird$. However, this is not the case if $C_{r_2}=\{\}$. Consider the program P':

$R_{P'} = \{r_1: fly. \quad r_2: \neg fly \leftarrow \sim bird.$ with $C_{r_2} = \{\}\}$ and $<_{\mathbf{R}} = \{\}$.

Then, the literal $\neg fly$ is not unfounded w.r.t. r_1 and Ø since there is no S to satisfy conditions in Def. 2.4. This, intuitively, implies that r_1 is blocked and fly is evaluated as unknown.

Example 2.2: (credit confusion problem) Consider the following expanded PEP, $P = <R_P, <_{\mathbf{R}}>$:

$R_P = \{$ /* If Ann is a foreign student (resp. teaching assistant) then she needs 12 (resp. 6) credits */

r_1: need_credits(ann,12)←foreign_stud(ann). $\quad$ r_2: need_credits(ann,6)←TA(ann).

r_3: TA(ann). $\quad$ r_4: foreign_stud(ann).

r_5:¬need_credits(ann,6)←need_credits(ann,12).

r'_5:¬need_credits(ann,12)←need_credits(ann,6).

with $C_{r_i} = \{\}$ for i=1,2,3,4, C_{r_5}={need_credits(ann,12)} and $C_{r'_5}$={need_credits(ann,6)}}

and $r_1 < r_5$, $r_2 < r_5$, $r_3 < r_5$, $r_4 < r_5$, $\quad$ $r_1 < r'_5$, $r_2 < r'_5$, $r_3 < r'_5$, $r_4 < r'_5$ and $r_1 < r_2$.

/* Rules r_5 and r'_5 have higher priority than the other rules */

The literal ¬TA(ann) is unfounded w.r.t. r_3 and Ø (in Def. 2.4 take *S*={¬TA(ann)}). So, TA(ann) can be reliably derived from rule r_3. Similarly, foreign_stud(ann) can be reliably derived from rule r_4. Since $r_1 < r_2$, the literal ¬need_credits(ann,6) is unfounded w.r.t. r_2 and Ø (in Def. 2.5 take *S*={¬need_credits(ann,6), need_credits(ann,12)}). So, need_credits(ann,6) can be reliably derived from rule r_2. In contrast, ¬need_credits(ann,12) is not unfounded w.r.t. r_1 and Ø. However, if *P'* is as *P* with $<_{\mathbf{R}}$={} then ¬need_credits(ann,6) is not unfounded w.r.t. r_2 and Ø in *P'*.

Definition 2.5 (truth valuation of a rule): Let *P* be a *PEP*. A rule *r* in *exp*(*P*) is *c-true* w.r.t. an interpretation *I* iff: (i) $I(Head_r) \geq I(Body_r)$ or (ii) $I(Body_r)$=1/2 and $I(\neg Head_r)$=1 or (iii) $I(Body_r)$=1 and ($I(Head_r)$=1/2 or $I(\neg Head_r)$=1) and $\neg Head_r$ is not unfounded w.r.t. *r* and *I*.

Definition 2.6 (c-model): Let *P* be a *PEP*. A consistent, coherent interpretation *I* of *P* is a *c-model* of *P* iff every rule in *exp*(*P*) is *c*-true w.r.t. *I*.

Example 2.3: Let *P* be as in Example 2.1 and *M* be a c-model of *P*. Then, *fly*∈*M* because r_1 is c-true w.r.t. *M* and ¬*fly* is unfounded w.r.t. r_1 and *M* ⊇Ø. In contrast, *fly* is not true in all models of *P'* of Example 2.1. The c-models of *P'* are M_1={~*bird*}, M_2=*coh*({*fly*,~*bird*}) and M_3=*coh*({¬*fly*,~*bird*}).

Example 2.4: Let *P* be as in Example 2.2. Then, *M*=*coh*({TA(ann), foreign_stud(ann), need_credits(ann,6), ¬need_credits(ann,12)}) is a c-model of *P*. We will show that *M* is the unique c-model of *P*. Let *M'* be a c-model of *P*. Then, ¬TA(ann), ¬foreign_stud(ann), ¬need_credits(ann,6), need_credits(ann,12) are unfounded w.r.t. *M*⊇Ø and rules r_3, r_4, r_2 and r'_5, respectively. Thus, *M*⊆*M'*. The literal need_credits(ann,12)∉*M'* because otherwise ¬need_credits(ann,6)∈*M'* (need_credits(ann,6) is unfounded w.r.t. r_5 and *M*⊇Ø) and thus, *M'* is contradictory.

Let *P* be a normal program and *I* an interpretation as defined in [Przy89, Przy90]. In [Przy90], a rule *r* is true w.r.t. *I* iff $I(Head_r) \geq I(Body_r)$. Since *P* is a normal program, rules do not contain classically negative literals. Consequently, the bodies of all contrapositives in *exp*(*P*) contain classically negative literals. This implies that all classically negative literals are unfounded w.r.t. any rule and *I*. If *I'* =*I*∪{~¬A|A is an atom of *P*} then conditions (ii) and (iii) in Def. 2.5 are not satisfied by *I'*, for all rules in *P*. This implies that a rule *r* in *P* is c-true w.r.t. *I'* iff *r* is true w.r.t. *I*.

Proposition 2.1: Let *P* be a normal program. *M* is a model of *P* iff *M*∪{~¬A|A is an atom of *P*} is a c-model of *P*.

3. Contradiction-Free Semantics

In this Section, we define the *contradiction-free model*, *stable c-models* and *contradiction-free semantics* of a *PEP*, *P*. We define the contradiction-free model of *P* as the least fixpoint of a monotonic operator and we show that it is the least grounded c-model of *P*.

Definition 3.1 ($\mathbf{W}_P$ operator): Let *P* be a *PEP* and *J* a set of literals. We define:

• $T_J(T)=\{L \mid \exists r{:}L \leftarrow L_1,...,L_n$ in $exp(P)$ s.t. (i) $L_i \in T \cup J, \forall i \leq n$ and (ii) $\neg L$ is unfounded w.r.t. r and $J\}$.

• $\mathbf{T}(J)=\cup\{T_J{}^{\uparrow a}(\emptyset) \mid a<\omega\}$, where ω is the first limit ordinal.

• $\mathbf{F}(J)$ is the greatest set of classical literals S s.t. $\forall L \in S$:

 If r is a rule in $exp(P)$ with $Head_r=L$ then $I(Body_r)=0$ or $\exists L' \in Body_r$ s.t. $L' \in S$.

• $\mathbf{W}_P(J)=coh(\mathbf{T}(J) \cup \sim\mathbf{F}(J))$.

When $\neg Head_r$ is not unfounded w.r.t. r and J, we say that r is blocked w.r.t. J. Note that the sequence $\{T_J{}^{\uparrow a}\}$ is monotonically increasing (w.r.t. $\subseteq$). So, $\mathbf{T}(J)$ is the least fixpoint of $\mathbf{T}_J$. We define the transfinite sequence: $I_0=\{\}$, $I_{a+1}=\mathbf{W}_P(I_a)$ and $I_a=\cup\{I_b \mid b<a\}$ if a is a limit ordinal.

Proposition 3.1: Let P be a PEP. $\{I_a\}$ is a monotonically increasing (w.r.t. $\sqsubseteq$) sequence of consistent, coherent interpretations of P.

Since $\{I_a\}$ is monotonically increasing (w.r.t. $\sqsubseteq$), there is a smallest countable ordinal d s.t. $I_d=I_{d+1}$.

Proposition 3.2: Let P be a PEP. Then, I_d is a c-model of P.

Definition 3.2 (contradiction-free semantics): Let P be a PEP. The *contradiction-free model* of P (CFM_P) is the c-model I_d. The *contradiction-free semantics* (CFS) of P is the "meaning" represented by CFM_P.

In Example 3.1, we show that contrapositives are necessary in order to avoid the derivation of complementary literals in $\{I_a\}$.

Example 3.1: Consider the PEP, $P=<R_P, <_{\mathbf{R}}>$:

$R_P=\{r_1{:}\ OK\_M.\ r_2{:}\ \neg rings.\ r_3{:}\ rings \leftarrow OK\_M.$ with $C_{r_3}=Body_{r_3}\}$ and $r_1<r_2<r_3$.

Rule r_3 expresses that if machine M is OK then it rings. Rule r_2 expresses the observation that machine M does not ring. Rule r_1 expresses the assumption that machine M is OK.

The program $exp(P)$ is as follows:

$R_{exp(P)}=\{r_1{:}\ OK\_M.\ r_2{:}\ \neg rings.\ r_3{:}\ rings \leftarrow OK\_M.$ $r'_3{:}\neg OK\_M \leftarrow \neg rings.$

with $C_{r_3}=Body_{r_3}$ and $C_{r'_3}=Body_{r'_3}\}$ and $r_1<r_2<r_3$, $r_1<r_2<r'_3$.

Since $rings$ is unfounded w.r.t. r_2 and $\emptyset$, $\neg rings \in \mathbf{T}(\emptyset)$. Since, the literal $\neg OK\_M$ is not unfounded w.r.t. r_1 and $\emptyset$, rule r_1 is blocked w.r.t. $\emptyset$. Since OK_M is unfounded w.r.t. r'_3 and $\emptyset$, $\neg OK\_M \in \mathbf{T}(\emptyset)$ (derived from r'_3). So, $\mathbf{T}(\emptyset)=\{\neg rings, \neg OK\_M\}$, $\mathbf{F}(\emptyset)=\{\}$ and $\mathbf{W}_P(\emptyset)=coh(\{\neg rings, \neg OK\_M\})$. Because $\mathbf{W}_P{}^{\uparrow 2}(\emptyset)=\mathbf{W}_P(\emptyset)$, it follows that $CFM_P = coh(\{\neg rings, \neg OK\_M\})$.

We will show that contrapositives are necessary in order to avoid the derivation of complementary literals. Assume that $exp(P)$=P. Then, $\neg OK\_M$ is unfounded w.r.t. r_1 and Ø and thus $OK\_M \in \mathbf{W}_P$(Ø). Consequently, *rings* is derived from r_3, since $\neg rings$ is unfounded w.r.t. r_3 and Ø. However, $\neg rings \in \mathbf{W}_P$(Ø) (derived from r_2), since *rings* is unfounded w.r.t. r_2 and Ø. So, $\mathbf{W}_P$(Ø)=$coh(\{OK\_M, rings, \neg rings\})$ which is inconsistent.

Let P' be as P with the additional rule r_4: *rings*. Let $r_3 < r_4$, expressing that r_4 is a more reliable observation than r_3. Then, $\mathbf{W}_{P'}$(Ø)=$coh(\{rings\})$ where *rings* is derived from rule r_4. Note that $\neg OK\_M$ is not unfounded w.r.t. r_1 and Ø in P' and thus r_1 is blocked w.r.t. Ø. In contrast, $\neg OK\_M$ is unfounded w.r.t. r_1 and $\mathbf{W}_{P'}$(Ø) because $Body_{r'3}$= $\neg rings$ is false w.r.t. $\mathbf{W}_{P'}$(Ø). Thus, $CFM_{P'} = \mathbf{W}_{P'}{\uparrow}^2$(Ø)=$coh(\{OK\_M, rings\})$.

Example 3.2: Let P be the program of Example 2.2. Then, CFM_P= coh({TA(ann), foreign_stud(ann), need_credits(ann,6), ¬need_credits(12)}). If P' is as P with $<_{\mathbf{R}}$={} then $CFM_{P'}$ =coh({TA(ann), foreign_stud(ann)}) which corresponds to the skeptical meaning of P'. If P' is as P with C_r =$Body_r$ $\forall r$ then ¬TA(ann) (resp. ¬foreign_stud(ann)) is not unfounded w.r.t. r_3 (resp. r_4) and Ø because of the contrapositive of r_2 (resp. r_1) in $exp(P)$ Thus, $CFM_{P'}$ ={}.

Proposition 3.3: Let P be a *PEP*. The complexity of computing CFM_P is $O(|HB_P|*|P|^2)$.

An algorithm for computing CFM_P is given in the Appendix A. The contradiction-free model of a *PEP* corresponds to the skeptical meaning of the program. Credulous meanings can be obtained using the transformation $P/_c I$, where I is an interpretation of P. The transformation P/I is defined in [GeLi88, Przy90] for a normal program P. $P/_c I$ extends P/I to *PEPs*.

Definition 3.3 (transformation $P/_c I$): Let P be an expanded *PEP* and I be an interpretation of it. The program $P/_c I$ is obtained as follows:

(i) Remove from P all rules that contain in their body a default literal ~L s.t. $I(L)$=1.
(ii) Remove from P any rule r with $I(\neg Head_r)$=1.
(iii) If r is a rule in P s.t. $I(Body_r)$=1 and $I(Head_r)$=1/2 then replace r with $Head_r \leftarrow \boldsymbol{u}$.
(iv) Remove from the body of the remaining rules of P any default literal ~L s.t. $I(L)$=0.
(v) Replace all remaining default literals ~L with $\boldsymbol{u}$.
(vii) Replace every classically negative literal $\neg A$ with a new atom $\neg\_A$.

Example 3.5: Let P be as in Example 2.2 and M be as in Example 2.4. Then
$P/_c I$ ={need_credits(6)←TA(mary). TA(mary). foreign_stud(mary).
¬need_credits(12)←need_credits(6). }

The program $P/_c I$ is a non-negative program with a special proposition $\boldsymbol{u}$. For any interpretation J, $J(\boldsymbol{u})=1/2$. When P is a normal program and M is a model of P [Przy90], $P/_c M \equiv P/M$ since Steps (ii), (iii), (vi) and (vii) do not have any effect on $P/_c M$. We say that a model M of P is the $least_v$ model of P iff $M(L) \leq M'(L)$ for any model M' and classical literal L of P. The $least_v$ model of P is the least fixed point of a monotonic operator given in [Przy90].

Definition 3.4 (grounded c-model): Let P be a *PEP* and M a c-model of P. M is a *stable c-model* of P iff $least_v(exp(P)/_c M)=M$.

Example 3.6: Let P' be as P in Example 2.2 with $<_{\mathbf{R}}=\{\}$. Let
$M_1=coh(\{$TA(ann), foreign_stud(ann), need_credits(ann,6), ¬need_credits(12)$\})$ and
$M_2=coh(\{$TA(ann), foreign_stud(ann), need_credits(ann,12), ¬need_credits(6)$\})$.
Then, M_1 and M_2 are stable c-models of P.
The program P of Example 2.2 has a unique stable c-model equal to CFM_P.

Let M be a stable c-model of P. Changing the ordering of the rules in P, the condition $least_v(exp(P)/_c M)=M$ is still be satisfied but M may not be a c-model of the new program. For example, let P be as in Example 2.2 and M be as in Example 2.4. Then, M is a stable c-model of P. If we replace $r_1<r_2$ in P with $r_2<r_1$, the condition $least_v(\exp(P)/_c M)=M$ is still satisfied. However, M is not a c-model of the new program P' because ¬need_credits(12) becomes unfounded w.r.t. r_1 and M and thus, r_1 is not c-true w.r.t. M in P'.

Proposition 3.4: Let P be a *PEP*. Then, CFM_P is a grounded c-model of P.

Proposition 3.5: Let P be a *PEP*. Then, CFM_P is the least grounded c-model of P.

4. Related Work

The contradiction-free semantics for *PEPs* is a generalization of the 3-*valued stable model semantics* which is defined for normal programs [Przy90].

Proposition 4.1: Let P be a normal program and M a set of classical literals. Then, M is a 3-valued stable model of P iff $M \cup \{\sim\neg A | A$ is an atom of $P\}$ is a grounded c-model of P.

Proposition 4.1 implies that the contradiction-free model of a normal program P coincides with the *well-founded model* (*WFM*) of P [vGRS91].

In [GeLi91], the *answer-set semantics* of an extended program is defined as the intersection of its answer-sets. However, the answer set semantics is not defined for all extended programs and can be contradictory. Moreover, the problem of finding whether an extended program has an answer set is NP-complete [Elkan90]. The following relationship between *CFS* and answer set semantics can be shown.

Proposition 42: Let P be an extended program. If $M \neq HB_P$ is an answer-set of P then $M \cup \{\sim A | A \notin M\}$ is a grounded c-model of P.

Prioritization of rules is investigated in [GLV91, LaVe92]. An *ordered logic program* is a partially-ordered set of rules without negation by default. Even though the *c-assumption-free semantics* [GLV91] and *assumption-free semantics* [LaVe92] are defined for all ordered logic programs, negation by default is not supported. The *skeptical c-partial model* of the program P in Example 3.1 is $\{OK\_M,\ rings\}$. According this model, machine M works OK and it rings even though rule r_2: $\neg rings \leftarrow$ has higher priority than rule r_1:$OK\_M \leftarrow$. This unintuitive result is derived because in [GLV91], the rule ordering $r' < r$ represents that rule r is an *exception* of rule r'. This corresponds in our frame work with the case that $C_r = \{\}\ \forall$ rule r. Indeed, let P' be as program P in Example 3.1 with $C_r = \{\}\ \forall$ rule r. Then, the contradiction-free model of P' equals the skeptical c-partial model $\{OK\_M,\ rings\}$. The skeptical c-partial model of $exp(P)$ (P expanded with contrapositives) is $\{\}$.

In [LaVe92], the *well-founded partial model* of an ordered logic program P is defined. Similarly to [GLV91], rule ordering in [LaVe92] represents *exceptions* and not *reliability*. This corresponds in our framework with the case that $C_r = \{\}\ \forall$ rule r. Another difference between the contradiction-free model and the well-founded partial model is demonstrated by the following example. The well-founded partial model of $P = \{p.\ \ \neg p \leftarrow q.\ \ \neg q.\ \ q.\}$ is $\{p\}$. According to this model, p is *true* even though $\neg p$ can also be derived from P. The reasoning in [LaVe92] is that the derivation of q and $\neg p$ is blocked and thus, p can be reliably derived. In [Stein89], a similar ambiguity blocking approach applied to inheritance networks was severely questioned. In our approach, ambiguities are propagated and thus rule $p \leftarrow$ is blocked. Note that $CFM_P = \{\}$, independently of the values of C_r. Proposition 4.3 gives the relationship of *CFS* with ordered logic.

Proposition 4.3: Let $P = \langle R_P, <_{\mathbf{R}} \rangle$ be a *PEP* which is free from default literals and $C_r = \{\}\ \forall$ rule r. Then, the set of classical literals in CFM_P is a subset of the well-founded partial.model of P [LaVe92] and coincides with the skeptical c-partial model of P [GLV91].

The use of contrapositives for resolving contradictions appears in [GiMa90, Witt91]. Yet, in these works, rule prioritization is not considered and $C_r = Body_r\ \forall$ rule r (all contrapositives of a rule are considered). Let P be a normal program with constraints. [GiMa90] defines a *generalized stable model* P as a 2-valued stable model [GeLi91] of the expansion of P with (i) contrapositives of rules and constraints and (ii) constraints $\{\bot \leftarrow L,\ \neg L \mid L \in HB_P\}$. Not all programs have a generalized stable model. We can show that if $P = \langle R_P, <_{\mathbf{R}} \rangle$ with $<_{\mathbf{R}} = \{\}$ and $C_r = Body_r\ \forall$ rule r then every *generalized stable model* of P is a stable c_model of P. However, the opposite is not valid.

Let P be a normal program with constraints. [Witt91] define the *strong belief revision model* (*SBRM*) of P as the *WFM* of the expansion of P with (i) contrapositives of rules and constraints, (ii) rules $\{L \leftarrow \sim\neg L \mid L \in HB_P\}$, and (iii) constraints $\{\bot \leftarrow L,\ \neg L \mid L \in HB_P\}$. Let $P = \langle R_P, <_{\mathbf{R}} \rangle$ with $<_{\mathbf{R}} = \{\}$ and $C_r = Body_r\ \forall$ rule r. We can show that if the rules $\{L \leftarrow \sim\neg L \mid L \in HB_P\}$

are removed from the expansion of P in [Witt91] and *coherence* is enforced in the computation of the *SBRM* then when the *SBRM* of P exists, it coincides with *CFS*.

5. Procedural Semantics

In this Section, we present *SLCF*-resolution (*CF* for contradiction-free), a proof procedure to answer queries on *PEPs* based on the contradiction-free semantics. *SLCF*-resolution is inspired by the approach to constructive negation taken in *SLDFA*-resolution [Drab93].

Substitutions in goals are replaced by constraints which are represented by Greek letters. The idempotent substitution $\{x_1/t_1,..., x_n/t_n\}$ corresponds to the constraint $x_1=t_1,...,x_n=t_n$. A constraint q is *satisfiable* iff $CET \models q$, where *CET* stands for the Clark equality theory [Lloy87]. We use the letter Q to represent a list of literals. A *goal* is a formula $\leftarrow q,Q$, where q is a satisfiable constraint. An *SLCF-refutation* of a goal is defined in Def. 5.1. Let $\leftarrow q,Q$ be a goal. $\mathbf{W}_P{\uparrow}^a(\emptyset) \models \exists q,Q$ iff there exists is an *SLCF*-refutation of rank a for the goal $\leftarrow q,Q$. Goals in an *SLCF*-refutation may contain a new type of literal $\sim_r c(L)$, where L is a classical literal and r is a rule. Intuitively, there exists an *SLCF*-refutation of rank a+1 for a goal $\leftarrow q,\sim_r c(L)$ iff $\exists q'$ s.t. q,q' is satisfiable and every ground instance of q,q',L is unfounded w.r.t. r and $\mathbf{W}_P{\uparrow}^a(\emptyset)$. A selection function selects a literal from a goal. The selected literal of a goal is underlined, e.g., the selected literal of $\leftarrow q,a,\underline{L},b$ is L.

Definition 5.1: Let P be a *PEP* and a be a countable ordinal. An *SLCF-refutation of rank* $a\geq 1$ for a goal G is a sequence of goals $G_1,...,G_n$ s.t. $G_1=G$, $G_n= \leftarrow q''$ and $\forall G_i$ one of the following is true:

1. $G_i= \leftarrow q,Q,\underline{L}(x_1,...,x_n),Q'$ and $\exists$ a variant r: $L(t_1,...,t_n)\leftarrow L_1,...,L_m$ of a rule in $exp(P)$ and
   $G_{i+1}= \leftarrow q,(x_1=t_1,...,x_n=t_n),Q, L_1,...,L_m, \sim_r c(\neg L(x_1,...,x_n)),Q'$.
2. $G_i= \leftarrow q,Q,\sim_r c(\underline{L}),Q'$ and $\exists q'$ s.t. $\leftarrow q,q',c(L)$ r-fails at rank $\leq a$ (Def. 5.3) and $G_{i+1}= \leftarrow q,q',Q,Q'$.
3. $G_i= \leftarrow q,Q,\sim\underline{L},Q'$ where L is a classical literal and one of the following is true:
   (i) there is q' s.t. $\leftarrow q,q',L$ fails at rank $< a$ (Def. 5.2) and $G_{i+1}= \leftarrow q,q',Q,Q'$.
   (ii) there is an *SLCF*-computed answer q' of rank $< a$ for $\leftarrow q,\neg L$ and $G_{i+1}= \leftarrow q,q', Q,Q'$.

The constraint q'' restricted to the free variables of G is an *SLCF-computed answer of rank a* for G.

Condition 1 expresses that the head L of a rule r is true if all literals in the body of the rule are true and $\neg L$ is unfounded w.r.t. r and *CFS*. The constraint q' in conditions 2 and 3(i) of Def. 5.1 can be computed similarly to the way failed answers are computed in [Drab91]. Note that an *SLCF*-computed answer of rank a+2 for $\leftarrow q,\sim L$ is q,q' iff (i) $\exists q'$ s.t. the goal $\leftarrow q,q',L$ fails at rank a+1 which means that every ground instance of $q,q',L \in \mathbf{F}(\mathbf{W}_P{\uparrow}^a(\emptyset))$ or (ii) $\exists q'$ s.t. q,q' is satisfiable and every ground instance of $q,q',\neg L \in \mathbf{W}_P{\uparrow}^{a+1}(\emptyset)$ (*coherence*).

Definition 5.2: Let P be a *PEP* and $a \geq 1$ a countable ordinal. A goal *G fails at rank a* iff there exists a tree T s.t. (i) T has root G, (ii) no node of T has the form $\leftarrow q$, and (iii) $\forall$ node N, N is a goal and

1. If $N = \leftarrow q, Q, \underline{L}(x_1, \ldots, x_n), Q'$ s.t. L is a classical literal then for every variant r: $L(t_1, \ldots, t_n) \leftarrow L_1, \ldots, L_m$ of a rule in $exp(P)$ s.t. $q, (x_1 = t_1, \ldots, x_n = t_n)$ is satisfiable, N has a child: $\leftarrow q, (x_1 = t_1, \ldots, x_n = t_n), Q, L_1, \ldots, L_m, Q'$.
2. If $N = \leftarrow q, Q, \sim L(x_1, \ldots, x_n), Q'$ s.t. L is a classical literal then one of the following is true:
   (i) There is an *SLCF*-computed answer q' of rank $< a$ for $\leftarrow q, L(x_1, \ldots, x_n)$ and N has children:
   $\leftarrow q, q_1, Q, \sim L(x_1, \ldots, x_n), Q'$, ..., $\leftarrow q, q_m, Q, \sim L(x_1, \ldots, x_n), Q'$, where $CET \models q \rightarrow q' \vee q_1 \vee \ldots \vee q_m$.
   (ii) N has a child: $\leftarrow q, Q, Q'$.

Let r be a rule and K a classical literal. A goal $\leftarrow q'', c(K)$ r-fails at rank $a+1$ iff $\exists q'$ s.t. q, q' is satisfiable and every ground instance of q, q', L is unfounded w.r.t. r and $\mathbf{W}_P{}^{\uparrow a}(\emptyset)$.

Definition 5.3: Let P be a *PEP*, $a \geq 1$ a countable ordinal and $G = \leftarrow q''$, $c(K)$ a goal where K is a classical literal. *G r-fails at rank a* iff there exists a tree T s.t. (i) T has root G (ii) no node of T has the form $\leftarrow q$, and (iii) $\forall$ node N, N is a goal and

1. If $N = \leftarrow q, Q, c(\underline{L}(x_1, \ldots, x_n)), Q'$ s.t. L is a classical literal then one of the following is true:
   (i) There is an *SLCF*-computed answer q' of rank $< a$ for $\leftarrow q, \neg L(x_1, \ldots, x_n)$ and N has children:
   $\leftarrow q, q_1, Q, c(L(x_1, \ldots, x_n)), Q'$, ..., $\leftarrow q, q_m, Q, c(L(x_1, \ldots, x_n)), Q'$, where $CET \models q \rightarrow q' \vee q_1 \vee \ldots \vee q_m$.
   (ii) For every variant r': $L(t_1, \ldots, t_n) \leftarrow L_1, \ldots, L_m$ of a rule in $exp(P)$ s.t. $q, (x_1 = t_1, \ldots, x_n = t_n)$ is satisfiable and $r' \nless r$, N has a child: $\leftarrow q, (x_1 = t_1, \ldots, x_n = t_n), Q, L'_1, \ldots, L'_m, Q'$, where:
   – if $L_i \in C_r$ and L_i is a classical literal then $L'_i = c(L_i)$,
   – if $L_i \in C_r$ and L_i is the default literal $\sim L'$ then $L'_i = c(\neg L')$,
   – if $L_i \notin C_r$ and L_i is a classical literal then $L'_i = L_i$,
   – if $L_i \notin C_r$ and L_i is the default literal $\sim L'$ then $L'_i = \neg L'$.
2. If $N = \leftarrow q, Q, \underline{L}(x_1, \ldots, x_n), Q'$ s.t. L is a classical literal then one of the following is true:
   – There is an *SLCF*-computed answer q' of rank $< a$ for $\leftarrow q, \neg L(x_1, \ldots, x_n)$ and N has children:
   $\leftarrow q, q_1, Q, L(x_1, \ldots, x_n), Q'$, ..., $\leftarrow q, q_m, Q, L(x_1, \ldots, x_n), Q'$, where $CET \models q \rightarrow q' \vee q_1 \vee \ldots \vee q_m$.
   – N has a child $\leftarrow q, Q, Q'$.

Proposition 5.1 (Soundness, Completeness): Let P be a *PEP*, R a fair selection rule, Q a list of classical and default literals and $G \leftarrow q, Q$ a goal. If q' is an *SLCF*-computed answer for G then every ground instance of q', Q is true w.r.t. CFM_P. If Qt is ground and true w.r.t. CFM_P then there is an *SLCF*-computed answer q' for G s.t. Qt is a ground instance of q', Q. Every ground instance of q, Q is false w.r.t. CFM_P iff the goal $\leftarrow q, Q$ fails.

Example 5.1: Consider the *PEP*, P=<R_P, $<_{\mathbf{R}}$ >:

R_P={r_1: $\neg q(0)$. r_2: $\neg q(1)$. r_3: $p(X)$. r_4: $\neg q(X) \leftarrow \sim s(X)$. r_5: $q(X) \leftarrow p(X)$.
with C_r=$Body_r$ $\forall$ rule r} and $r_1 < r_3$. Then, the expanded program *exp*(P) is as follows:

$R_{exp(P)}$={r_1: $\neg q(0)$. r_2: $\neg q(1)$. r_3: $p(X)$. r_4: $\neg q(X) \leftarrow \sim s(X)$. r'_4: $s(X) \leftarrow q(X)$.
r_5: $q(X) \leftarrow p(X)$. r'_5: $\neg p(X) \leftarrow \neg q(X)$. with C_r=$Body_r$ $\forall$ rule r} and $r_1 < r_3$.

An *SLCF*-refutation for $\leftarrow q(X)$ is: $G_1 = \leftarrow \underline{q}(X)$, $G_2 = \leftarrow \underline{p}(X), \sim_{r5} c(\neg q(X))$ (using rule r_5 in condition 1 of Def. 5.1), $G_3 = \leftarrow \sim_{r3} c(\neg p(X)), \sim_{r5} c(\neg q(X))$ (using rule r_3 in condition 1),
The goal $\leftarrow X \neq 1, c(\neg p(X))$ r_3-fails since there exists a tree satisfying the conditions of Def. 5.3.

$\leftarrow X \neq 1, c(\neg p(X))$
| (using rule r'_5 in condition 1(ii) of Def. 5.3)
$\leftarrow X \neq 1, c(\neg \underline{q}(X))$

Note that, since $r_1 < r_3$, rule r_1 does not satisfy the condition 1(ii) of Def 5.3.

Since $X \neq 1$, $X=1$ is unsatisfiable, rule r_2 does not satisfy the condition 1(ii) of Def 5.3.

| (using rule r_4 in condition 1(ii) of Def. 5.3)
$\leftarrow X \neq 1, c(\neg s(X))$ (it is a leaf)

$G_4 = \leftarrow X \neq 1, \sim_{r5} c(\neg q(X))$ (condition 2 of Def. 5.1),
The goal $\leftarrow X \neq 1, c(\neg q(X))$ r_5-fails since there exists a tree satisfying the conditions of Def. 5.3.

$\leftarrow X \neq 1, c(\neg \underline{q}(X))$
| (using rule r_4 in condition 1(ii) of Def. 5.3)
$\leftarrow X \neq 1, c(\neg \underline{s}(X))$ (it is a leaf)

$G_5 = \leftarrow X \neq 1$ (condition 2 of Def 5.1),
So, an *SLCF*-computed answer for $\leftarrow q(X)$ is $X \neq 1$.

SLCF-resolution is an ideal procedural semantics since termination is not guaranteed for all programs. A proof procedure for propositional logic programs that incorporates loop detection and always terminates is given in Appendix B.

6. Conclusions

We have presented a new semantics for *prioritized extended programs* (*PEP*). The semantics, called *contradiction-free semantics* (*CFS*), is a generalization of the well-founded semantics for normal programs [vGRS91] and defined for all *PEPs*. We describe fixpoint, model theoretic and procedural characterizations of *CFS*. The *contradiction-free model* of a program P is the least grounded c-model of P and it represents the skeptical "meaning" of P. Grounded c-models of P represent credulous "meanings" of P. The degree of "skepticism" in *CFS* depends on the contrapositive sets of the program rules. If *P* is an ordered logic program then the *CFS* of *P* coincides with the *skeptical c-partial model* of *P* [GLV91] and is a subset of the *well-founded partial model* of *P* [LaVe92]. When the Herbrand base of a *PEP* is finite, the complexity of computing *CFS* of *P* is polynomial w.r.t. $|P|$.

APPENDIX A: Computation of CFM_P

Let *P* be a propositional *PEP*. The algorithm ***CFM***(program *P*) returns the contradiction-free model of *P*. To compute **F**(*I*) and the set of unfounded literals w.r.t. a rule *r* and *I*, the complement set is constructed first, as in [vGRS91]. The complexity of the algorithm is O(HB_P, $|P|^2$).

```
CFM(program P)
{ new_I={};
  repeat
     I=new_I;
    repeat                /* compute T(I) */
      for each rule r of exp(P) do
         if new_I(Body_r)=1 and unfounded(¬Head_r, r, I) then add Head_r to new_I;
      end /* for */
    until no change in new_I;

    compl_F={};
    repeat                /* compute F(I) */
      for each rule r of exp(P) do
       if I(Body_r)≠0 and all body classical literals of r are true w.r.t I then add Head_r to compl_F;
      end /* for */
    until no change in compl_F;
    for each L∈HB_P do
       if L∉compl_F then add ~L to new_I;
     end /* for */

    new_I = coh(new_I);
  until I=new_I;
  return(I);
}

unfounded(literal L, rule r, interpretation I)   /* returns TRUE is L is unfounded w.r.t. r and I */
{  compl_U={};
   repeat
```

```
        for each rule $r'$ of $exp(P)$ s.t. $r' \not< r$ do
          if $I(Body_{r'}) \neq 0$ and for each $L \in Body_{r'}$ either $L$ is true w.r.t. $compl\_U$ or $L \notin C_{r'}$
          then add $Head_{r'}$ and $\sim \neg Head_r$ to $compl\_U$;
        end /* for */
    until no change in $compl\_U$;
    if $L \notin compl\_U$ then return($TRUE$); else return($FALSE$);
}
```

APPENDIX B: Propositional Proof Theory

Let P be a propositional *PEP* and L a literal. *CFS*_deriv(L,{}) returns SUCC (resp. FAIL) iff L is true (resp. false or unknown) w.r.t. *CFS*. The routine distance_in_list(L, *Literal_list*, *Distance*) returns SUCC if the input $Literal\_list = L_1,..., L_i, L, L_{i+1},..., L_n$. The output *Distance* is ZERO if $L,L_{i+1},...,L_n$ are default literals or $L,L_{i+1},...,L_n$ are of the form c(K).

```
CFS_deriv(classical literal $L$, ancestor list $Anc$)
{ if $L$ unifies with $\sim\sim L'$ or $\neg\neg L'$ or $\sim\neg L'$ then return($CFS$_deriv($L'$, $\{L\}\cup Anc$));
  if $\sim L \in Anc$ then return (FAIL); endif
  if distance_in_list($L$, $Anc$, $Dist$)=SUCC then
      if $Dist$= ZERO and $L$ is a default literal then return(SUCC); else return(FAIL); endif
  endif
  if $L$ unifies with $\sim L'$ and $CFS$_deriv($\neg L'$, $\{L\}\cup Anc$)=SUCC then return(SUCC);
  endif
  if $L$ unifies with $\sim L'$ then  /* $L$ is a default literal */
      for every rule $r$: $L' \leftarrow Body_r \in exp(P)$ do
          if not($\exists K \in Body_r$ s.t. $CFS$_deriv($\sim K$, $\{L\}\cup Anc$)=SUCC) then return(FAIL); endif
      endfor
      return(SUCC);
  else                                /* $L$ is a classical literal */
      flag=RETRY;
      for every rule $r$: $L \leftarrow Body_r \in exp(P)$ and if flag=RETRY do
          flag=SUCC;
          for every literal $K \in Body_r$ do
              if $CFS$_deriv($K$, $\{L\}\cup Anc$)=FAIL then flag=RETRY; break; endif
          endfor
      endfor
      if flag=SUCC and unfounded($\neg L$, $r$, $\{L\}\cup Anc$)=SUCC then return(SUCC); else
return(FAIL);
      endif
  endif
}
```

```
unfounded(literal $L$, rule label $r$, ancestor list $Anc$)
/* return $SUCC$ if $L$ is unfounded w.r.t. $r$ and $CFS$ */
{
  if $L$ unifies with $\neg\neg L'$ then return(unfounded($L'$, $r$, $Anc$));
  if $L$ unifies with $\sim L'$ then return(unfounded($\neg L'$, $r$, $Anc$));
  if in_ancestor($c(L)$, $Anc$, $Dist$)=SUCC then
      if $Dist$=ZERO then return(SUCC); else return(FAIL); endif
  endif
  for every rule $r$: $L \leftarrow Body_r$ do
      if not ($\exists K \in C_r$ s.t. unfounded($K$, $r$, $\{c(L)\} \cup Anc$)=SUCC or
              $\exists K \in Body_r$ such that $CFS$_deriv($\sim K$, $\{c(L)\} \cup Anc$)=SUCC)
      then return(FAIL);
      endif
  endfor
  return(SUCC);
}
```

REFERENCES

[BrLi91] S. Brass, U.W. Lipeck, "Semantics of Inheritance in Logical Object Specifications", Proc. of the 2nd Intern. Conference on Deductive and Object-Oriented Databases, 1991, pp.411-430.

[BrLi93] S. Brass, U.W. Lipeck, "Bottom-Up Query Evaluation with Partially Ordered Defaults", Proc. of the 3rd Intern. Conference on Deductive and Object-Oriented Databases (DOOD'93), 1993.

[Drab91] W. Drabent, "What is failure? An approach to constructive negation", Technical Report LiTH-IDA-R-91-23, Linkoping University, August 91, Submitted to *Acta Informatica*.

[Drab93] W. Drabent, "SLS-resolution without floundering", Proc. of the 2nd Intern. Workshop on Logic programming and Non-monotonic Reasoning, 1993, pp.82-98.

[DuRu91] P.M. Dung, P. Ruamviboonsuk, "Well-Founded Reasoning with Classical Negation", First Intern. Workshop on Logic Programming and Non-monotonic Reasoning, 1991, pp.120-132.

[Dung93] P.M. Dung, "An Argumentation Semantics for Logic Programs with Explicit Negation", Proc. of the 10th Intern. Conference on Logic programming (ICLP'93), 1993, pp.616-630.

[Elkan90] C. Elkan, "A Rational Reconstruction on Nonmonotonic Truth Maintenance System", Artificial Intelligence, Elsevier Science Publishers, 43(2), 1990, pp. 219-234.

[Fitt85] M. Fitting, "A Kripke-Kleene Semantics for Logic Programs", Journal of Logic Programming, 2(4), 1985, pp. 295-312.

[GLV91] D. Gabbay, E. Laenens, D. Vermeir, "Credulous vs. Sceptical Semantics for Ordered Logic Programs", Proc. of the 2nd Intern. Conference on Knowledge Representation and Reasoning (KR'91), Morgan Kaufmann, 1991, pp. 208-217.

[GeVe91] P. Geerts, D. Vermeir, "Credulous and autoepistemic reasoning using ordered logic", First Intern. Workshop on Logic Programming and Non-monotonic Reasoning, 1991, pp.21-36.

[GeLi88] M. Gelfond, V. Lifschitz, "The Stable Model Semantics for Logic Programming", Proc. of the 5th Symposium on Logic Programming, MIT Press, 1988, pp. 1070-1080.

[GeLi91] M. Gelfond, V. Lifschitz, "Classical Negation in Logic programs and Disjunctive Databases", New Generation Computing, 9, OHMSHA, 1991, pp. 365-385.

[Hunt92] A. Hunter, "A Conceptualization of Preferences in Non-monotonic Proof Theory", D. Pearce and G. Wagner (eds). Logics in AI, LNCS 633, 1992, pp. 174-188.

[LaVe90] E. Laenens, D. Vermeir, "A Fixpoint Semantics for Ordered Logic", Journal of Logic and Computation, 1(2), Oxford University Press, 1990, pp. 159-185.

[LaVe92] E. Laenens, D. Vermeir, "Assumption-free Semantics for Ordered Logic Programs: On the Relationship Between Well-founded and Stable Partial Models", Journal of Logic and Computation, 2(2), Oxford University Press, 1992, pp. 133-172.

[LeRu92] N. Leone, P. Rullo, "Stable Model Semantics and its Computation for Ordered Logic Programs", Proc. of the 10th European Conf. on Artificial Intelligence (ECAI'92), 1992, pp. 92-96.

[Lloy87] J. W. Lloyd, "Foundations of Logic Programming", Springer-Verlag, second edition, 1987.

[PAA91*a*] L.M. Pereira, J.J. Alferes, J.N. Aparicio, "Contradiction Removal within Well Founded Semantics", A. Nerode, W. Marek, V.S. Subrahmanian (eds.), Proc. of the 1st Intern. Workshop on Logic programming and Non-monotonic Reasoning, 1991, pp.106-119.

[PAA91*b*] L.M. Pereira, J.N. Aparicio, J.J. Alferes, "Nonmonotonic Reasoning with Well Founded Semantics", Proc. of the Intern. Conference on Logic Programming (ICLP',91), 1991, pp.475-504.

[PAA92] L.M. Pereira, J.J. Alferes, J.N. Aparicio, "Contradiction Removal with Explicit Negation", Proc. of the Applied Logic Conference, ILLC, 1992.

[PeAl92] L.M. Pereira, J.J. Alferes, "Well Founded Semantics for Logic Programs with Explicit Negation", Proc. of the 10th European Conference on Artificial Intelligence, 1992, pp. 102-106.

[PAA93] L.M. Pereira, J.N. Aparicio, J.J. Alferes, "Nonmonotonic Reasoning with Logic Programming", Journal of Logic Programming, Elsevier Science Publishing, 17, 1993, pp.227-263.

[Przy90] T. Przymusinski, "Well-Founded Semantics Coincides with the Three-Valued Stable Semantics", Fundamenta Informaticae XIII, IOS Press, 1990, pp. 445-463.

[Stein89] L. A. Stein, "Skeptical Inheritance: Computing the Intersection of Credulous Extensions", Proc. of the 11th Intern. Joint Conference on Artificial Intelligence, 1989.

[vGRS91] A. van Gelder, K.A. Ross, J.S. Schlipf, "The Well-Founded Semantics for General Logic Programs", Journal of the Association for Computing Machinery", 38(3), July 1991, pp.620-650.

[Witt91] C. Witteveen, "Skeptical Reason Maintenance is Tractable", Proc. of the 2nd Intern. Conference on Knowledge Representation and Reasoning (KR91), 1991, pp.570-581.

[Witt92] C. Witteveen, "Expanding Logic Programs", Logics in AI, LNCS 633, 1992, pp. 372-390.

[Wagn93] G. Wagner, "Reasoning with Inconsistency in Extended Deductive Databases ", Proc. of the 2nd Intern. Workshop on Logic programming and Non-monotonic Reasoning, 1993, pp.300-315.