A Classification Theory of Semantics of Normal Logic Programs: I. Strong Properties

Jürgen Dix
University of Koblenz-Landau
Department of Computer Science
Rheinau 1,
56075 Koblenz, Germany

Abstract. Our aim in this article is to present a method for classifying and characterizing the various different semantics of logic programs with negation that have been considered in the last years. Instead of appealing to more or less questionable intuitions, we take a more structural view: our starting point is the observation that all semantics induce in a natural way non-monotonic entailment relations " $\mid\!\sim$ ". The novel idea of our approach is to ask for the properties of these $\mid\!\sim$ -relations and to use them for describing all possible semantics. The main properties discussed in this paper are adaptations of rules that play a fundamental rôle in general non-monotonic reasoning: *Cumulativity* and *Rationality*. They were introduced and investigated by Gabbay, Kraus, Lehmann, Magidor and Makinson. We show that the 3-valued version COMP_3 of Clark's completion, the stratified semantics M_P^{supp} as well as the well-founded semantics WFS and two extensions of it behave very regular: they are *cumulative*, *rational* and one of them is even *supraclassical*. While Pereira's recently proposed semantics O-SEM is not *rational* it is still *cumulative*. *Cumulativity* fails for the regular semantics REG-SEM of You/Yuan (recently shown to be equivalent to three other proposals). In a second article we will supplement these *strong* rules with a set of *weak* rules and consider the problem of uniquely describing a given semantics by its strong and weak properties together.

1 Introduction

This article is the first in a series of three. The first two articles are devoted to *normal* programs while the third treats *disjunctive* programs. The methods and techniques (as well as some of the results) introduced here and in the second article are fundamental and will be extended in the third paper to disjunctive programs.

We begin this introduction with some general remarks about the history of the problem (Section 1.1), and present in Section 1.2 our own approach. The organization of the paper is given in Section 1.3.

1.1 Some historical remarks

Historically, semantics for logic programs have been considered in the logic programming community for about 20 years. It began with [CKPR73, Kow74, vEK76] and led to the definition and implementation of *PROLOG*, a by now theoretically well-understood programming language (at least the declarative part consisting of Horn-clauses: *pure PROLOG*). Extensions of PROLOG allowing negative literals have been also considered in this area: they rely on the idea of *negation-as-finite-failure*, we call them *Logic-Programming-semantics* (or shortly *LP-semantics*).

In parallel, starting at about 1980, *Non-monotonic Reasoning* entered into computer science and began to constitute a new field of active research. In recent years, independently of the research in logic programming, people interested in non-monotonic reasoning also tried to define declarative semantics for programs containing negative literals in their bodies (or even *disjunctions* in their heads). They defined various semantics by appealing to (different) intuitions they have about programs. This second line of research started in 1986 with the *Workshop on the Foundations of Deductive Databases and Logic Programming* organized by Jack Minker: the revised papers of the proceedings were published in [Min88]. The *stratified* (or the similar *perfect*) semantics presented there can be seen as a splitting-point: it is still of interest for the logic programming community (see [CL89]) but its underlying intuitions were inspired by non-monotonic reasoning. Semantics of this kind leave the philosophy underlying classical logic programming in that their primary aim is not to model negation-as-finite-failure, Clark's completion or SLDNF-resolution, but to construct new, more powerful semantics suitable for applications of some forms of non-monotonic reasoning. Let us call such semantics *NMR-semantics*.

Nowadays, due to the work of Apt, Blair and Walker, Fitting, Lifschitz, Przymusinski and others, very close relationships between these two independent research lines became evident. Methods from logic programming, e.g. least fixpoints of certain operators, were used successfully to define NMR-semantics. The most important NMR-semantics are the stable semantics STABLE ([BF91, GL88]) and the well-founded semantics WFS ([vGRS88]). First attempts to extend WFS by allowing *reasoning by cases* are due to Minker and his group ([Min93, LMR92]) and to Schlipf ([Sch92]).

While all NMR-semantics coincide on the class of *stratified* programs (which is a proper subclass of all programs) with the single two-valued model M_P^{supp} (defined by Apt, Blair and Walker in [ABW88]), they behave quite differently on non-stratified programs. Thus the question of *the most canonical extension of* M_P^{supp} *for the class of all programs* arises. The number of different approaches already suggests that there is no such single candidate and that the *optimal* (or best suited) semantics depends on the area of application.

Recently three interesting overviews about negation in logic programming have appeared: Minker's article in the *Special Issue of the Journal of Logic Programming on Non-Monotonic Reasoning and Logic Programming* ([Min93]), Apt and Bol's article in the jubileum (10th anniversary) issue of the *Journal of Logic Programming* ([AB94]) and the author's article in the *Proceedings of the Konstanz Colloquium in Logic and Information* ([Dix95]). While Minker's article gives an almost complete description of all the activities and different approaches in the field, Apt/Bol and Dix also try to present recent research results in a comprehensive and detailed manner. Although no proofs are given, all important definitions and notions are formally introduced to illustrate the underlying ideas in a precise and strict fashion. The articles of Apt/Bol and Dix can be seen as complementary in a sense: while Apt and Bol concentrate more on *LP-semantics*, Dix is more concerned with *NMR-semantics*.

1.2 Our approach

In this series of papers we intend to develop a framework which makes it possible to obtain results of the form

Any semantics satisfying certain properties is uniquely determined by these.

Having carefully investigated the recent approaches we discovered an irregular behaviour of some semantics. The reason is that in many cases one tried to improve a given semantics by putting an additional mechanism on top of its definition. This additional mechanism was often motivated by only one single program that, according to the intuitions of the respective group of researchers, was not handled correctly by the given semantics. We noticed that the new semantics sometimes have more serious shortcomings than the original semantics and therefore tried to find principles where all semantics should be checked against.

Our approach is partially inspired by the work of Kraus, Lehmann, Magidor and Makinson in general non-monotonic reasoning ([Mak89, Mak94, KLM90, LM92, DM92]). They abstracted from particular (propositional versions of) non-monotonic logics, such as *Default Logic*, *Autoepistemic Logic* and *Circumscription* and developed a general (*proof* and *model*) theory for non-monotonic relations " $\mid\!\sim$ " together with *soundness* and *completeness* results.

Following their approach, we will first associate to any semantics SEM a sceptical non-monotonic consequence relation SEM^{scept} and then axiomatically present two types of abstract properties of this relation:

- The first type, called *strong* principles, are adaptions of some of the properties introduced by Kraus, Lehmann, Magidor and Makinson: they have nothing to do with our special setting of logic programs but nevertheless will turn out to be very useful. They will be investigated in this article and also used to distinguish between some of the LP-semantics.

- The second type, called *weak* principles, reflect the specific idea of *negation-as-failure* in logic programming: the two clauses "$a \leftarrow \neg b$" and "$b \leftarrow \neg a$" are, viewed as *logic programs*, completely different, but viewed as classical formulae, they are equivalent. The first program states that b is false (because there is no clause with b in its head) and therefore a is true, while the second program states that a is false and b true. These principles are defined and investigated in the second paper. We argue that any semantics should be checked against these properties. In fact, all our properties were inspired by irregular behaviour of some of the existing semantics.

We claim that by taking both types of principles together, *weak* and *strong* properties can be used to uniquely characterize certain semantics. The corresponding representation conjectures will be stated and discussed in the second article.

Figure 1 may help to illustrate the different classes of programs (together with semantics defined for them) that we are considering: while we are concerned in the first two articles with *normal* programs (the lower half of the diagram) we will extend our methods in the last article to disjunctive programs (the upper half of the diagram).

Partial results of this and the second article have already appeared in preliminary versions (without proofs) in the extended abstracts [Dix91a, Dix91b, Dix92a] as well as in the author's PhD-thesis [Dix92c].

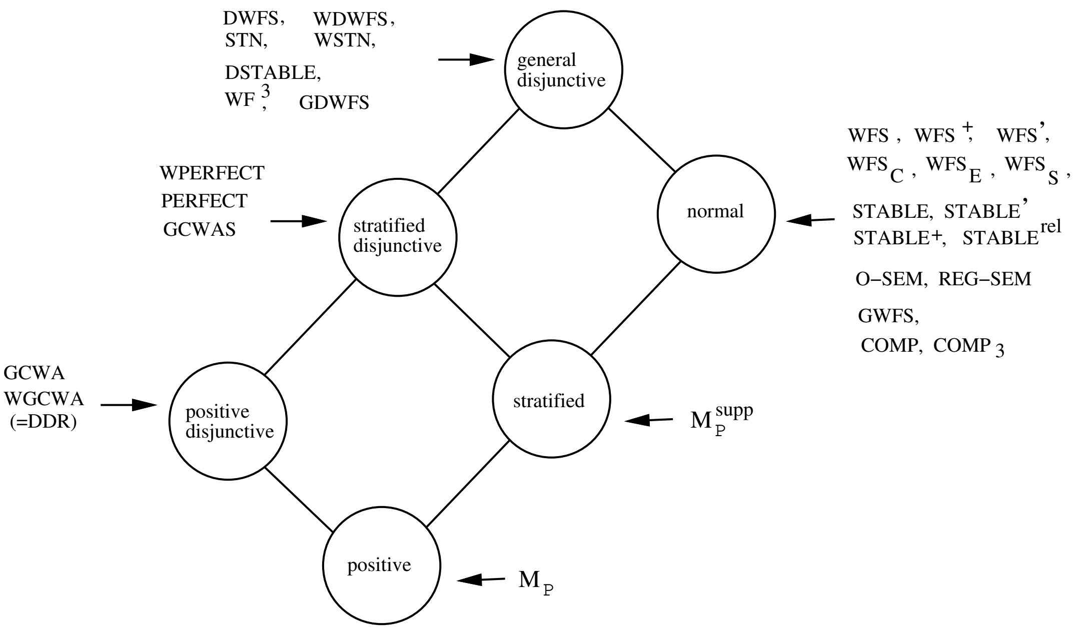

Figure 1: Classes of Programs

1.3 Organization of the paper

In Section 2, we fix the standard terminology that will be used in this series of papers and introduce our notion of a semantics SEM and its induced *sceptical version* SEM^{scept}. Section 3 presents the *LP-view* (COMP and COMP$_3$) and the stronger *NMR-view* (extensions of M_P^{supp}) of logic programs together with its abstract properties. We investigate in Section 4 the well-founded semantics WFS and construct two interesting extensions of it (WFS$^+$ and WFS$'$). We also consider Pereira's O-SEM as well as the regular semantics REG-SEM defined by You and Yuan and determine the properties of all these semantics. Section 5, finally, ends with some concluding remarks.

2 SEM and its sceptical version SEM^{scept}

In Section 2.1 we give some standard definitions for the context of logic programs, for our use of three-valued logic and for other notions used throughout this series of papers. The main results of this paper state that certain semantics satisfy certain abstract properties. These properties will be introduced in Section 2.2 and briefly illustrated with some standard non-monotonic theories. In Section 2.3 we give a very general definition of a semantics SEM and show how any semantics SEM induces a *sceptical entailment* SEM^{scept}. We end this section with a comparison of the abstract rules in general and their adaptation to our specialized setting of logic programs (Section 2.4).

2.1 Logic programs and three-valued logic

A *general disjunctive* logic program consists of a finite number of *rules* that allow arbitrary *positive* clauses to appear in their heads:

$$A_1 \vee \ldots \vee A_n \leftarrow B_1, \ldots, B_m, \neg C_1, \ldots, \neg C_l \quad \text{where } n \geq 1 \text{ .}$$

t | **f** Lattice **2** Ordering: $\leq$	**t** | **u** | **f** Lattice $\mathbf{3}_t$ Ordering on *truth*: $\leq_t$	**t** **f** \\ / **u** Partial Lattice $\mathbf{3}_k$ Ordering on *knowledge*: $\leq_k$

Figure 2: Important (partial) lattices of truth-values

If $l = 0$, the program is *positive disjunctive*; if $n = 1$, the program is *normal*; if $n = 1$ and $l = 0$, the program is *positive* (or *definite*): see Figure 1.

All the literals may contain free variables: the rules are viewed as shorthand notations for all possible instantiations. P_{inst} stands for the (infinite) set of fully instantiated rules and facts (*ground* clauses). A program is, in addition, called *propositional* or *Datalog*, if it does not contain any function symbols, but only propositional variables or their negations. Note that by taking the ground instantiation P_{inst} of P, we can view P as a propositonal, but in general *infinite* program.

We denote the Herbrand base with respect to a program P by $B_{\mathcal{L}_P}$ or simply by B_P: the underlying language $\mathcal{L}_P$ is given by the symbols in P. $Th(\Phi)$ denotes the classical *deductive closure* of the set of formulae Φ and Fml denotes the set of all formulae. Finally, let MIN-MOD(T) denote the class of all *two-valued* minimal Herbrand models of an arbitrary theory T (not necessarily a logic program). A Herbrand model $\mathcal{A}$ of T is called *minimal*, if there is no other model $\mathcal{A}'$ of T such that for all atoms a of the Herbrand base B_T: $\mathcal{A}' \models a$ implies $\mathcal{A} \models a$.

We also need some notions from 3-valued logic. We use truth values **t** "true", **f** "false", **u** "undefined" and the Kleene connectives $\vee$, $\wedge$, $\neg$ and $\leftarrow$. $\leftarrow$ is the *weak* implication, where "$\mathbf{u} \leftarrow \mathbf{u}$" is considered to be *true*. Additionally, we can use two different orderings of the truth values: the lattice $\mathbf{3}_t$ defined by $f \leq_t \mathbf{u} \leq_t t$ (*truth-ordering*) and the semi-lattice $\mathbf{3}_k$ defined by $\mathbf{u} \leq_k \mathbf{t}$, $\mathbf{u} \leq_k \mathbf{f}$ (*knowledge-ordering*): see Figure 2. We regard a three-valued interpretation $\mathcal{I}$ as a pair $\langle T; F\rangle$, consisting of the sets of atoms T (the *true* atoms) and F (the *false* atoms). We are using True($\mathcal{I}$) (resp. False($\mathcal{I}$)) to denote the atoms that are true (resp. false) in $\mathcal{I}$ and we will also represent $\mathcal{I}$ by just enumerating its ground literals: $\mathcal{I} = \text{True}(\mathcal{I}) \cup \{\neg x : x \in \text{False}(\mathcal{I})\}$. It is clear that an interpretation $\mathcal{I}$ on atoms can be uniquely extended to an interpretation $\hat{\mathcal{I}}$ of all sentences. The relations $\leq_t$ and $\leq_k$ also naturally extend. Thus, to define a mapping from the set of three-valued interpretations into itself, it suffices to consider mappings from $\mathbf{3}^{B_P}$ into itself. In the sequel, we write "$\mathbf{3}_k^{B_P} \rightarrow \mathbf{3}_k^{B_P}$" to indicate that we are interested in the $\leq_k$-ordering.

Any semantics based on a two-valued theory Σ can be seen as a three-valued interpretation $\mathcal{I}$=$\langle$T;F$\rangle$ with T=$\{A : A$ a ground atom with $\Sigma \models A\}$ and F=$\{A : A$ a ground atom with $\Sigma \models \neg A\}$. But note that in general $\mathcal{I}$ may no longer be a three-valued *model* of Σ[1].

Definition 2.1 (Dependency-Graph $\mathcal{G}_P$)
For a logic program P, the dependency graph $\mathcal{G}_P$ *is a finite directed graph whose vertices are the predicate symbols from P. There is a* positive *(resp.* negative*) edge from R to R'* :iff *there is a clause in P with R in its head and R' occurring positively (resp. negatively) in its body.*

[1] Take for example Σ to be $Th(\{a \leftarrow \neg b, p \leftrightarrow (a \vee b)\})$.

We also say

- R depends on R' *if there is a path in $\mathcal{G}_P$ from R to R' (by definition, R depends on itself),*
- R depends positively on R' *if there is a path in $\mathcal{G}_P$ from R to R' containing only positive edges (by definition R depends positively on itself),*
- R depends negatively on R' *if there is a path in $\mathcal{G}_P$ from R to R' containing at least one negative edge.*

A generalization of $\mathcal{G}_P$ is the infinite instantiated dependency graph $\mathcal{G}_P^{inst}$ *whose vertices are the elements of $B_{\mathcal{L}_P}$. The edges are defined analogously: instead of P one takes P_{inst}.*

2.2 Kraus, Lehmann, Magidor and Makinson's rules

Let $\mathcal{L}$ be the classical language of propositional logic: We have the usual notion of classical entailment $Th(\Phi)$; for $\phi \in Th(\{\mathbf{t}\})$ we write $\models \phi$. The following structural properties for an entailment relation "$\mathrel{|\!\sim}$" *between single formulae* were considered by Kraus, Lehmann and Magidor:[2]

Right Weak.:	$\models \alpha \rightarrow \beta$	and	$\gamma \mathrel{\vert\!\sim} \alpha$	$imply$	$\gamma \mathrel{\vert\!\sim} \beta.$
Reflexivity:				Let	$\alpha \mathrel{\vert\!\sim} \alpha.$
And:	$\alpha \mathrel{\vert\!\sim} \beta$	and	$\alpha \mathrel{\vert\!\sim} \gamma$	$imply$	$\alpha \mathrel{\vert\!\sim} \beta \wedge \gamma.$
Or:	$\alpha \mathrel{\vert\!\sim} \gamma$	and	$\beta \mathrel{\vert\!\sim} \gamma$	$imply$	$\alpha \vee \beta \mathrel{\vert\!\sim} \gamma.$
Left Log. Equiv.:	$\models \alpha \leftrightarrow \beta$	and	$\alpha \mathrel{\vert\!\sim} \gamma$	$imply$	$\beta \mathrel{\vert\!\sim} \gamma.$
Cautious Monotony:	$\alpha \mathrel{\vert\!\sim} \beta$	and	$\alpha \mathrel{\vert\!\sim} \gamma$	$imply$	$\alpha \wedge \beta \mathrel{\vert\!\sim} \gamma.$
Cut:	$\alpha \mathrel{\vert\!\sim} \beta$	and	$\alpha \wedge \beta \mathrel{\vert\!\sim} \gamma$	$imply$	$\alpha \mathrel{\vert\!\sim} \gamma$
Rationality:	$not\ \alpha \mathrel{\vert\!\sim} \neg\beta$	and	$\alpha \mathrel{\vert\!\sim} \gamma$	$imply$	$\alpha \wedge \beta \mathrel{\vert\!\sim} \gamma.$
Negation Rat.	$\alpha \mathrel{\vert\!\sim} \beta$	$implies$	$\alpha \wedge \gamma \mathrel{\vert\!\sim} \beta$	or	$\alpha \wedge \neg\gamma \mathrel{\vert\!\sim} \beta.$
Disjunctive Rat.:	$\alpha \vee \beta \mathrel{\vert\!\sim} \gamma$	$implies$	$\alpha \mathrel{\vert\!\sim} \gamma$	or	$\beta \mathrel{\vert\!\sim} \gamma.$

Kraus, Lehmann and Magidor also defined a *model-theory* for "$\mathrel{|\!\sim}$": they considered the class of all models, with an additional partial ordering on it, and defined a satisfiability relation. They termed this framework a *model preference logic Q*. Results of their work are various *representation theorems* for different axiom systems, such as **C** consisting of *Right Weakening*, *Reflexivity*, *And*, *Left Logical Equivalence*, *Cautious Monotony* and *Cut*; **P** consisting of the rules of **C** and the rule *Or* and finally **R** consisting of **P** and the rule *Rationality* (see [KLM90] and [LM92]). It can be verified that, if we assume the rules of the system **C**, the following implications are strict:

$$\textit{Rationality} \implies \textit{Disjunctive Rationality} \implies \textit{Negation Rationality}$$

Cut is one of the most natural conditions and is satisfied for all existing non-monotonic formalisms. Makinson (see his overview article [Mak94]) used a mapping[3] C: $2^{Fml} \rightarrow 2^{Fml}$ to define the following *infinitistic* versions of the rules of **P**: (Φ, Ψ are arbitrary sets of formulae)

[2]"$\mathrel{|\!\sim}$" can be extended to a relation between *finite* sets of formulae using $\wedge$.

[3]He calls it a *closure-operation*.

Supraclassicality: $Th(\Phi) \subseteq C(\Phi)$,

Distributivity: $C(\Phi) \cap C(\Psi) \subseteq C(\, Th(\Phi) \cap Th(\Psi)\,)$,

Cumulativity: $\Phi \subseteq \Psi \subseteq$ C(Φ) implies C(Φ) = C(Ψ).

When considering infinite sets of formulae (for example an instantiated logic program), Makinsons terminology seems more appropriate than " $\mid\!\sim$ ". Of course, the two notions " $\mid\!\sim$ " and C are connected via $\Psi \subseteq C(\Phi)$ **:iff** $\Phi \mid\!\sim \Psi$ (see [Mak94] for a comparison of these two approaches). If we denote by C_{fin} the restriction of C to *finite* sets, we get the following

Lemma 2.2 (C versus $\mid\!\sim$)

a) *If* And *holds:* Cumulativity *(for C_{fin})* $\simeq$ Cautious Monotony *and* Cut.

b) *If* Cut *holds:* Supraclassicality *(for C_{fin})* $\simeq$ Reflexivity *and* Right Weakening.

c) Distributivity *(for C_{fin})* $\simeq$ Or *and* Left Logical Equivalence.

Proof: We only verify a) (both b) and c) can be proved along the same lines). *Cumulativity* splits into two implications:

$$\Phi \subseteq \Psi \subseteq C(\Phi) \text{ implies } C(\Phi) \subseteq C(\Psi), \tag{1}$$

$$\Phi \subseteq \Psi \subseteq C(\Phi) \text{ implies } C(\Psi) \subseteq C(\Phi). \tag{2}$$

Using our connection between C and $\mid\!\sim$, these two equations translate to

$$\text{If } \Phi \subseteq \Psi,\ \Phi \mid\!\sim \Psi \text{ then: } \quad \text{for all formulae } \gamma \ \ (\Phi \mid\!\sim \gamma \text{ implies } \Psi \mid\!\sim \gamma), \tag{3}$$

$$\text{If } \Phi \subseteq \Psi,\ \Phi \mid\!\sim \Psi \text{ then: } \quad \text{for all formulae } \gamma \ \ (\Psi \mid\!\sim \gamma \text{ implies } \Phi \mid\!\sim \gamma). \tag{4}$$

Let C_{fin} satisfy Cumulativity. To prove *Cautious Monotony* and *Cut* for $\mid\!\sim$ we simply set $\Phi := \{\alpha\}$ and $\Psi := \{\alpha, \beta\}$.

Now let $\mid\!\sim$ satisfy *Cautious Monotony* and *Cut*. To prove the implications (1) and (2) for C_{fin} we can represent Φ and Ψ as conjunctions of their elements (this is justified because *AND* holds) and we are done. ■

From now on we will also use the term *Cumulativity* for relations $\mid\!\sim$. To illustrate *Cumulativity* and *Rationality*, we consider the *Closed World Assumption* CWA and Circumscription CIRC. We define CWA syntactically as a closure operation and CIRC semantically as a set of models of a given theory T:

- CWA(T)=$Th(T \cup \{\neg\phi_{pos} : \ \phi_{pos}$ a positive formula with not $T \vdash \phi\})$,
- CIRC(T)=MIN-MOD(T).

We write $circ(T)$ for the corresponding set of formulae $\{\phi : \ \text{CIRC}(T) \models \phi\}$. Both CWA and CIRC induce in an obvious way $\mid\!\sim$ -relations:

$$T \mid\!\sim_{\text{CWA}} \phi \ \ \underline{\textbf{:iff}} \ \ \phi \in \text{CWA}(T) \ \text{ and } \ T \mid\!\sim_{\text{CIRC}} \phi \ \ \underline{\textbf{:iff}} \ \ \phi \in circ(T).$$

CWA was one of the first non-monotonic closure operations. It is sometimes too strong: CWA($\{\phi \vee \psi\}$) is inconsistent (if neither ϕ, ψ nor their negations are tautologies and ϕ, ψ are independent of each other).

Let us now restrict ourselves to *universal* theories T. It is easily seen that CIRC(T)$\neq \emptyset$, so that CIRC avoids CWA's inconsistencies for those theories T. The example just mentioned shows that CWA(T) does not satisfy *Cautious Monotony*: from CWA($\{\phi \vee \psi\}$) we can derive ϕ and $\neg\phi$ (because the theory is inconsistent), but if we add ϕ, $\neg\phi$ is no longer derivable (CWA($\{\phi \vee \psi,\ \phi\}$) $\neq Fml$).

The following lemma, well-known in non-monotonic reasoning, shows that CIRC is cumulative, as is CWA for consistent theories:

Lemma 2.3 (Properties of CWA and CIRC)

a) *CWA satisfies* Cut.

b) *If CWA*(Φ) *is consistent, then* $\Phi \subseteq \Psi \subseteq$ *CWA*(Φ) *implies CWA*$(\Phi)\ \subseteq\ $*CWA*$(\Psi)$.

c) *CIRC is cumulative but not rational.*

2.3 SEM and SEM^{scept}

We now consider the problem of adapting the " $\mid\!\sim$ "-formalism to logic programs:

- Proof-theoretically, semantics SEM$_P(U)$ of a program P together with a set U of atoms, can be defined as a set of literals that are derivable from P and U for a particular derivation mechanism (such as SLDNF-resolution).

- Model-theoretically, all existing semantics can be defined as subsets of MOD$_{3-val}(P)$ (the set of all three-valued models of P). More precisely we can use the Herbrand models with respect to the language $\mathcal{L}_P$: SEM$_P\ \subseteq\ MOD_{3-val}^{Herb_{\mathcal{L}_P}}(P)$.

Leaving disjunctive semantics aside (the reader is referred to the (forthcoming) third article in this series or to [Dix92b, DM94a]), we will consider in the first two papers of this series the following semantics

LP-semantics: Clark's completion *comp* ([Cla78]) and its three-valued variants SEM$_{\text{Fitting}}$ ([Fit85]) and SEM$_{\text{Kunen}}$ ([Kun87]).

NMR-semantics: The least Herbrand model M_P for *definite* programs, the supported Herbrand model M_P^{supp} ([ABW88]) for *stratified* programs and the following semantics defined for all programs: STABLE ([GL88, BF91]), STABLE$^+$ (defined in the second article), STABLE$_C$ ([Sch92]), STABLE^{rel} ([DM94b, DM94c]), WFS ([vGRS88, vGRS91]), WFS$^+$ and WFS$'$ ([Dix92a]), WFS$_C$ ([Sch92]), WFS$_E$ ([CK91]), WFS$_S$ ([HY91]), GWFS ([BLM90]), O-SEM ([PAA92]), and the regular semantics REG-SEM ([YY90]) that was recently proved to be equivalent (see [YY93]) to Sacca and Zaniolo's *partial models* ([SZ90, SZ91]), to Przymusinski's $\leq_k$*-maximal 3-valued stable models* ([Prz91, Prz90]) and to Dung's *preferred extensions* ([Dun91]).

Definition 2.4 (SEM)
A semantics SEM is a mapping from the class of all programs into the powerset of the set of all 3-valued Herbrand structures. SEM assigns to every program P a set of 3-valued Herbrand models of P:

$$SEM_P\ \subseteq\ MOD_{3-val}^{Herb_{\mathcal{L}_P}}(P).$$

We will from now on use the notation $P \cup U$ to represent a logic program: U is a distinguished set of *atoms* (no nontrivial program clauses are contained in U). P still may contain clauses with empty bodies. We also use the more terse $\text{SEM}_P(U)$ instead of $\text{SEM}_{P\cup U}$. This definition already indicates a fundamental difference to the general " $\mid\!\sim$ "-framework: our $\mid\!\sim_P$ is not defined between *arbitrary* program clauses.

Definition 2.5 (Sceptical entailment relation $\mid\!\sim_P$)
Let P be a program and U a set of atoms. Any semantics SEM induces a sceptical entailment relation SEM^{scept} as follows:

$$SEM^{scept}(U) := \bigcap_{\mathcal{M}\in \text{SEM}_P(U)} \{L : L \text{ is a pos. or neg. literal with: } \mathcal{M} \models L\}$$

Comparing with the " $\mid\!\sim$ " framework of Kraus, Lehmann and Magidor, we can equivalently define a " $\mid\!\sim_P$"-relation between sets of atoms *U (positive literals) on the left hand side, and sets of arbitrary* literals *X on the right hand side:*

- $u_1 \wedge \ldots \wedge u_n \mid\!\sim x_1 \wedge \ldots \wedge x_m$ **:iff** $\{x_1, \ldots, x_m\} \subseteq SEM_P^{scept}(\{u_1, \ldots, u_n\})$

If we are just interested in deriving *ground* literals from a logic program P, we use the notation $\text{GCWA}(P) = \{L : L \text{ a ground literal with MIN-MOD}(T) \models L\}$ for the corresponding sceptical semantics. The induced sceptical relation $\mid\!\sim_{\text{GCWA}}$ is now defined for a fixed program P and is a relation between sets[4] of *atoms* and sets of *literals*

$$(u_1 \wedge \ldots \wedge u_n) \mid\!\sim_{\text{GCWA}} (x_1 \wedge \ldots \wedge x_m) \quad \textbf{:iff} \quad \{x_1, \ldots, x_m\} \subseteq \text{GCWA}(P \cup \{u_1, \ldots, u_n\})$$

The following example shows that even in this restricted case of propositional programs, *Rationality* (which is a stronger version of *Cautious Monotony* as we illustrate in the next section) is not fulfilled for CIRC:

Example 2.6 (CIRC is not rational)

$$\begin{array}{llll} P_{CIRC}: & p & \leftarrow & \neg b \\ & b & \leftarrow & c \\ & c & \leftarrow & p, \neg a \end{array}$$

MIN-MOD(P_{CIRC})=$\{\{p,a\}, \{b\}\}$, i.e. "<u>not</u> GCWA(P_{CIRC}) $\mid\!\sim \neg p$", "GCWA(P_{CIRC}) $\mid\!\sim \neg c$", and MIN-MOD($P_{CIRC} \cup \{p\}$)=$\{\{p,a\}, \{p,c,b\}\}$, therefore "<u>not</u> GCWA($P_{CIRC} \cup \{p\}$) $\mid\!\sim \neg c$".

Remark 2.7 (Extending $\mid\!\sim_P$ to a relation between sets of literals)
We can easily extend our $\mid\!\sim_P$-entailment in a uniform way to a relation between arbitrary consistent sets of literals by setting

$$SEM_P^{scept}(X) := SEM_{P^*}^{scept}(U),$$

where $X = U \cup \neg V$ (U, V sets of atoms) and P^* is the program obtained from P by just deleting all clauses "$a \leftarrow$ body" with $\neg a \in X$. We define the underlying language of P^* to be $\mathcal{L}_P$: atoms in P should not get lost by cancelling some clauses. This is an important point and will be discussed in the second article (in the definition of P reduced by M *and at the end of Section 5.2).*

We call these generalizations of our rules extended. *I.e. the* Extended Cut *is the Cut viewed as a relation between arbitrary consistent sets of literals as defined above.*

[4] viewed as conjunctions

However, as pointed out by Li-Yan Yuan, this extension is not harmless. I.e. it is **not** the case that if some of our structural properties (defined in this or in the subsequent article) hold for a semantics in our original setting (where no negative literals are allowed on the left hand side of $\mid\!\sim_P$) then they also hold under this broader view (where such negative literals are allowed). Even for the weakest condition, the *Cut*, we can give a counteraxample. Yuan came up with the following program

$$\begin{array}{rrcl} P_{ext\_Cut}: & a & \leftarrow & b \\ & b & \leftarrow & \neg a \end{array}$$

The semantics WFS$^+$ that will be introduced in Section 4.8, derives a and $\neg b$ from the above program. If we apply the Cut to $P_{ext\_Cut} \cup \{\neg b\}$ we get

$$WFS^+(P_{ext\_Cut} \cup \{\neg b\}) \subseteq WFS^+(P_{ext\_Cut}).$$

But $P_{ext\_Cut} \cup \{\neg b\} = \{a \leftarrow b\}$ and WFS$^+(\{a \leftarrow b\}) = \{\neg a, \neg b\}$, which is not a subset from $\{a, \neg b\}$. This example shows, that the *Extended Cut* adds some new restrictions. Whether one wants this or not depends perhaps on the application or on the possibility to incremental compute a semantics. It seems that for these purposes, the extended Cut is a nice property. In fact, most semantics satisfying the (*basic*) *Cut* (resp. *Cumulativity*) also satisfy the *extended Cut* (resp.*extended Cumulativity*): e.g. WFS, STABLE, REG-SEM.

Note that SEM$_P^{scept}(U)$ itself can be seen as a three-valued model of $P \cup U$. With this notion, we are able to introduce two kinds of *extensions* of semantics. When we say that a semantics SEM$'$ *extends* another semantics SEM, we have to distinguish between two different notions:

- SEM $\leq_k$ SEM$'$: this means that SEM$'$ classifies more atoms as true or false than SEM, or
- SEM$'$ is defined for a class of programs that strictly includes the class of programs for which SEM is defined and for all programs of this smaller class, the two semantics coincide.

The first notion also makes perfectly sense for semantics defined for the same class of programs.

2.4 KLM-rules in our specialized setting

The greatest difference of our $\mid\!\sim_P$-setting to the general $\mid\!\sim$ -framework is that we do not have " $\mid\!\sim$ " (or C) as a relation on the *whole* set of formulae. Indeed this would not make sense because any notion of a *derived* program clause (with respect to a given semantics) leads to very simple counterexamples of our properties. Take for example the program $a \leftarrow \neg b$ with the canonical model $\{a, \neg b\}$. The clause $b \leftarrow \neg a$ is true in this model (therefore derivable) but added to the program, it drastically changes its meaning. Even more tricky definitions of *derivable clauses* lead to similar counterexamples.

A second difference to the KLM-framework is that we interpret the rules as *infinitistic*: our definition of entailment is a relation between *infinite* sets of atoms and literals

$$\mid\!\sim_P \subseteq 2^{\text{Atoms}} \times 2^{\text{Literals}}.$$

As indicated above, we can easily generalize this to $\mid\!\sim_P \subseteq 2^{\text{Literals}} \times 2^{\text{Literals}}$ in a straightforward way (but then we get *stronger* rules). In our setting the rules of *Right Weakening*, *Reflexivity*, *And*, and *Left Logical Equivalence* are completely trivial and always satisfied. Therefore the only interesting rules are *Cumulativity* (*Cautious Monotony* and *Cut*)

Cumulativity: If $U_1 \subseteq U_2 \subseteq \mathrm{SEM}_P^{scept}(U_1)$, then $\mathrm{SEM}_P^{scept}(U_1) = \mathrm{SEM}_P^{scept}(U_2)$.

and a strengthened form of *Cautious Monotony*

Rationality: If $U_1 \subseteq U_2,\ U_2 \cap \{A : \mathrm{SEM}_P^{scept}(U_1) \models \neg A\} = \emptyset$, then $\mathrm{SEM}_P^{scept}(U_1) \subseteq \mathrm{SEM}_P^{scept}(U_2)$.

Rationality is in any *sceptical* semantics a stronger form of *Cautious Monotony* because "$\alpha \mid\!\sim \beta$" implies "not $\alpha \mid\!\sim \neg\beta$".

Remark 2.8 (Finite versus Infinite)
It turns out that for our abstract properties Cautious Monotony*,* Cut *or* Rationality *there is no difference between formulating it for* infinite*,* finite *or even* one element *sets U_2. This is obvious by the structure of the rule: if* Cautious Monotony *(resp.* Cut*) holds for* one element *sets U_2 then it also holds for* finite *sets U_2. The step from finite to infinite sets is not so obvious but all semantics justify it. For* Rationality *even the step from* one element *sets to finite sets is not so obvious. But again there are no counterexamples.*

When we consider disjunctive programs we will extend $\mid\!\sim$ to a relation between sets of *positive disjunctions* and sets of *pure disjunctions* (i.e. sets of disjunctions consisting solely of positive or negative literals) or even arbitrary disjunctions. In this extended setting the two remaining rules of *Or* and *Disjunctive Rationality* make sense (note the disjunction $\vee$ before the $\mid\!\sim$ -sign) and turn out to be very interesting.

Let us end this section with some remarks concerning the proof theory developed in [KLM90]. Kraus, Lehmann and Magidor prove that certain additional rules can be derived using some others. For example they show that the *Loop*

$$\frac{\alpha_0 \mid\!\sim \alpha_1, \quad \alpha_1 \mid\!\sim \alpha_2, \ldots, \alpha_{k-1} \mid\!\sim \alpha_k, \quad \alpha_k \mid\!\sim \alpha_0}{\alpha_0 \mid\!\sim \alpha_k} \quad \text{for } k \in \mathit{IN} \qquad (\mathit{Loop})$$

is derivable in the system **P**. Makinson noted that the *Converse of Or* (this is the rule "$(\alpha \vee \beta) \mid\!\sim \gamma$ implies $\alpha \mid\!\sim \gamma$ and $\beta \mid\!\sim \gamma$" [5] in conjunction with *Left Logical Equivalence* already implies *Monotony*: if $\alpha \mid\!\sim \gamma$ then $(\alpha \wedge \beta) \mid\!\sim \gamma$. In our restricted setting, however, this is not true: we will see in the third paper that Przymusinski's *perfect semantics* (PERFECT) for stratified disjunctive programs satisfies all the rules of **P** but not the *Loop* (which already fails for WFS) and that Minker's *weak generalized closed world assumption* (WGCWA) for positive disjunctive programs satisfies *Converse of Or* without being monotone.

3 LP-semantics versus NMR-semantics

In this section we review the *LP-viewpoint* and the *NMR-viewpoint* of logic programs. While the first is closely related to the procedural SLDNF-resolution (Section 3.1) the latter can be seen as an attempt to define stronger semantics by detecting loops (Section 3.2). For the class of *stratified* programs a unique two-valued Herbrand model M_P^{supp} can be constructed (Section 3.3).

[5] It turns out that this property is useful to distinguish between certain disjunctive semantics (see [Dix92b]).

3.1 SLDNF, COMP and COMP$_3$

Quite independently of the work in non-monotonic reasoning, there evolved interesting work in the logic programming community: proof and model theory for *normal* logic programs (programs, where negative literals are allowed in the body) were defined. The proof theory originated from the well-known SLD-resolution of definite programs; it is based on the notion of *negation-as-finite-failure*: The precise definitions of SLDNF-*resolution*, *tree*, etc. are very complex: see [Llo87, Apt90] for the exact definitions[6]. In order to get an intuitive idea, it is sufficient to describe the following underlying principle:

Principle 3.1 (A "naive" SLDNF-resolution)
If in the construction of an SLDNF-tree a negative literal $\neg L_{ij}$ is selected in the list $\mathcal{L}_i = \{L_{i1}, L_{i2}, \ldots\}$, then we try to prove L_{ij}.
If this fails finitely *(it fails because the generated subtree is finite and failing), then we take $\neg L_{ij}$ as proved and we go on to prove $L_{i(j+1)}$.*
If L_{ij} succeeds, then $\neg L_{ij}$ fails and we have to backtrack to the list $\mathcal{L}_{i-1}$ of preliminary subgoals (the next rule is applied: "backtracking").

The corresponding semantics COMP is defined by Clark's completion $comp(P)$ (see [Cla78]). In logic programming, much work aimed at finding (syntactically defined) classes of programs, for which *soundness* and – even more importantly – *completeness* results hold (see [DC90, CL89] and [Stä94]).

The idea of Clark was that a program P consists not only of the implications, but also of the information that *these are the only ones.* Roughly speaking, he argues that one should interpret the "$\leftarrow$"-arrows in rules as "$\leftrightarrow$"-arrows. We do not give the exact definitions here, as they are very complex; in the non-propositional case, a symbol for equality, together with axioms describing it has to be introduced. Let us show with an example that COMP is not cumulative (*Cut* holds but *Cautious Monotony* fails):

Example 3.2 (COMP is not cautious monotonic)

$$
\begin{array}{lrcl}
P_{comp}: & a & \leftarrow & \neg p \\
 & a & \leftarrow & b \\
 & p & \leftarrow & a \\
 & p & \leftarrow & b \\
 & b & \leftarrow & b
\end{array}
\qquad
\begin{array}{lrcl}
comp(P_{comp}): & a & \leftrightarrow & \neg p \vee b \\
 & p & \leftrightarrow & a \vee b \\
 & b & \leftrightarrow & b
\end{array}
$$

$comp(P_{comp})$$=Th(\{\ a \leftrightarrow (\neg p \vee b),\ p \leftrightarrow (a \vee b),\ b \leftrightarrow b\ \}) = Th(\{a, b, p\})$. *But $comp(P_{comp} \cup \{p\})$ is the conjunction of $a \leftrightarrow (\neg p \vee b)$ and $p \leftrightarrow (a \vee b \vee \mathbf{t})$, so it is exactly $Th(\{p,\ a \leftrightarrow b\})$.*

Note that $comp(P)$ may become inconsistent: $comp(p \leftarrow \neg p) = Th\{p \leftrightarrow \neg p\}) = Fml$ but the failure of Cumulativity does not depend on it. Fitting ([Fit85]) solved this inconsistency problem by considering the following $\leq_k$-monotone operator

$$\Phi_P : \mathbf{3}_k{}^{B_P} \longmapsto \mathbf{3}_k{}^{B_P};\ \mathcal{I} \longmapsto \Phi_P(\mathcal{I})$$

[6]Quite recently, there are also attempts to improve these classical notions: see [AB94].

$$\Phi_P(\mathcal{I})(A) = \begin{cases} \mathbf{t}, \text{ if there is a clause } A \leftarrow L_1, \ldots, L_n \text{ in } P_{inst} \text{ with:} \\ \quad \text{– } \forall i \leq n \text{ we have:} \\ \quad\quad * \ \mathcal{I}(L_i) = \mathbf{t}. \\ \mathbf{f}, \text{ if for all clauses } A \leftarrow L_1, \ldots, L_n \in P_{inst} \text{ we have:} \\ \quad \text{– } \exists i \leq n \text{ with:} \\ \quad\quad * \ \mathcal{I}(L_i) = \mathbf{f}. \\ \mathbf{u}, \text{ otherwise.} \end{cases}$$

Fittings approach seems to be very elegant: just take the least fixpoint of Φ_P as the intended *canonical* three-valued model of P. This fixpoint is the $\leq_k$-least Herbrand model of P (we use the ordering $\mathbf{u} \leq_k \mathbf{t},\ \mathbf{u} \leq_k \mathbf{f}$ on the truth values!). He also defined a three-valued formulation $comp_3(P)$ of the completion that has this fixpoint as its $\leq_k$-least Herbrand model. On the other hand, Kunen defined a variant that is recursive enumerable: it corresponds to truth in all models, not only Herbrand models, of $comp_3(P)$.

Lemma 3.3
Let $U_1 \subseteq U_2$ be (possibly infinite) sets of ground atoms.

a) *For $U_2 \cap False(lfp(\Phi_{P\cup U_1})) = \emptyset$, we have:*

$$\mathcal{I} \leq_k \mathcal{I}' \ \textit{ implies } \ \Phi_{P\cup U_1}(\mathcal{I}) \leq_k \Phi_{P\cup U_2}(\mathcal{I}').$$

b) *For $U_2 \subseteq True(lfp(\Phi_{P\cup U_1}))$, we have:*

$$\mathcal{I} \leq_k lfp(\Phi_{P\cup U_1}) \ \textit{ implies } \ \Phi_{P\cup U_2}(\mathcal{I}) \leq_k lfp(\Phi_{P\cup U_1}).$$

Proof: The proof proceeds by the construction of the least three-valued Herbrand model of $comp_3$ as the least fixpoint of Φ_P. It is similar to the proof of Lemma 4.1 (see Section 4.1) and can be found in [Dix91b]. ■

Using this lemma we are able to show that the three-valued versions of COMP are not only cumulative but also rational:

Theorem 3.4 (Cumulativity and Rationality of COMP$_3$)

a) *The Fitting-semantics SEM$_{Fitting}$, given by the $\leq_k$-least three-valued Herbrand model (or, equivalently, by all Herbrand models) of comp$_3$, is cumulative and rational.*

b) *The Kunen-semantics SEM$_{Kunen}$, given by all three-valued models of comp$_3$, is cumulative and rational.*

Proof: **a)** This is just the application of the previous lemma. To prove *Rationality* we use a). Let the assumptions of *Rationality* be fulfilled, i.e.

$$U_1 \subseteq U_2,\ U_2 \cap \{A :\ lfp(\Phi_{P\cup U_1}) \models \neg A\} = \emptyset.$$

This means that the assumptions of a) are satisfied. We can therefore apply and iterate the lemma using $\emptyset \leq_k \emptyset$ and get

$$lfp(\Phi_{P\cup U_1}) \leq_k lfp(\Phi_{P\cup U_2})$$

To prove the *Cut* we use b) of the previous lemma. Let $U_1 \subseteq U_2 \subseteq lfp(\Phi_{P\cup U_1})$. The assumptions of b) are therefore satisfied: we can apply and iterate the lemma using $\emptyset \leq_k lfp(\Phi_{P\cup U_1})$ and get

$$lfp(\Phi_{P\cup U_2}) \leq_k lfp(\Phi_{P\cup U_1}).$$

b) The above lemma remains true if one replaces lfp($\Phi_{P\cup U_1}$) by $\Phi_{P\cup U_1}\!\uparrow^{\omega}$. Considering truth with respect to ground literals, we have: $\Phi_P\!\uparrow^{\omega}=\bigcup_{n\in IN} \Phi_P\!\uparrow\! n$. The underlying language does not matter.

■

3.2 NMR-Intuition compared with LP-Intuition

The close relationship between COMP and SLDNF (for which COMP was invented) and its weakness from the NMR-view (due to the different treatment of loops) is best shown by the following:

Example 3.5 (COMP vs. NMR)

$$\begin{array}{rlll} P_{NMR}: & p & \leftarrow & p \\ & q & \leftarrow & \neg p \end{array} \qquad \begin{array}{rlll} P'_{NMR}: & p & \leftarrow & p \\ & q & \leftarrow & \neg p \\ & r & \leftarrow & \neg r \end{array}$$

$$\begin{array}{rlll} comp(P_{NMR}): & p & \leftrightarrow & p \\ & q & \leftrightarrow & \neg p \end{array} \qquad \begin{array}{rlll} comp(P'_{NMR}): & p & \leftrightarrow & p \\ & q & \leftrightarrow & \neg p \\ & r & \leftrightarrow & \neg r \end{array}$$

?-q:	*No (COMP).*	*?-p*:	*Yes (COMP).*
	Yes (NMR).		*No (NMR).*

For both programs, the answers of the completion-semantics do not match our NMR-intuition! In the case of P_{NMR} we expect q to be derivable, since we expect $\neg p$ to be derivable: the only possibility to derive p is the rule $p \leftarrow p$ which, obviously, will never succeed. But $q \notin Th(\{q \leftrightarrow \neg p\}) = comp(P_{NMR})$! In the case of P'_{NMR} we expect p not to be derivable, for the same reason: the only possibility to derive p is the rule $p \leftarrow p$. But $p \in Fml = Th(\{r \leftrightarrow \neg r\}) = comp(P'_{NMR})$!

Note that the answers of the completion-semantics agree with the mechanism of SLDNF: $p \leftarrow p$ represents a loop. The completion of P' is inconsistent: this led Fitting to consider the three-valued version of $comp(P)$ introduced in the last section. This approach avoids the inconsistency (the query ?-p is not answered "yes") but it still does not answer "no" as we would like to have.

The last example motivates the need for a semantics that improves COMP and shows the difficulties in obtaining one: loops should be detected. In general, this problem is undecidable because the halting problem reduces to it. This is also the reason why the stable or well-founded semantics are, in the predicate logic case, not recursive (not even recursive enumerable): both are Π^1_1-complete. In addition, these two extensions of SLDNF do not agree on *which* loops should be detected. Let us underline the following idea that is worked out and investigated at length in [Bol90, Bol91, Bol93]:

- *NMR-Semantics = SLDNF + Loop-check*

3.3 M_P^{supp}

The immediate consequence operator $T_P : 2^{B_P} \longmapsto 2^{B_P}; \mathcal{I} \longmapsto T_P(\mathcal{I})$ can be defined in a straightforward way for *normal* programs:

$$T_P(\mathcal{I})(A) = \begin{cases} \mathbf{t}, \text{ if there is a clause } A \leftarrow L_1, \ldots, L_n \in P_{inst} \\ \quad \text{with:} \\ \qquad \text{– } \forall i \leq n \text{ we have:} \\ \qquad\qquad * \ \mathcal{I}(L_i) = \mathbf{t}, \\ \mathbf{f}, \text{ otherwise,} \end{cases}$$

but the properties "T_P is monotone and continuous" and "there exists a least Herbrand model $M_P = T_P{\uparrow}^{\omega} = lfp(T_P)$" get lost.

The problem in applying T_P to an arbitrary program is that at a stage i an atom A may still be false (so that it gives rise to derive new positive atoms via a clause of the form $A_{new} \leftarrow \neg A$) but that on a later stage $j > i$ the atom A may become true, so that all previously drawn inferences have to be rejected: this is exactly caused by the non-monotonicity of T_P!

Note that while $comp_3$ assigns r the value **u** (undefined) in the program P'_{NMR} (which makes sense), it also assigns **u** to p in P_{NMR}. The problems with these positive and negative cycles can be avoided for an interesting large class of programs: the idea is to rule out *all* programs having a cycle (not only a direct negative link) with a negative edge in their dependency graph, as for example the program $P_{unstrat}$: $p \leftarrow \neg q$, $q \leftarrow \neg p$. Programs without such cycles (called *stratified*) such as P_{strat}: $p \leftarrow \neg q$, $q \leftarrow \neg b$, induce a natural priority ordering on their relation-symbols. If, however, the clause $b \leftarrow p$ is added, the resulting program is no more stratified: there is a cycle with a negative edge between p and p. Equivalently, stratification may be formulated as follows

Definition 3.6 (Stratification)
A program P is called stratified*, if it is decomposable as $P = P_1 \cup \ldots \cup P_n$, such that the following holds (for i=1, ...,n):*

1. *If a relation-symbol* R *occurs* positively *in a clause in P_i, then all clauses containing* R *in their heads are contained in $\bigcup_{j \leq i} P_j$.*
2. *If a relation-symbol* R *occurs* negatively *in a clause in P_i, then all clauses containing* R *in their heads are contained in $\bigcup_{j < i} P_j$.*

We say: "P is stratified via $P_1 \cup \ldots \cup P_n$" and call every P_i "stratum of P".

We can now define the ordinal power $T_P^{\omega} : 2^{B_P} \longmapsto 2^{B_P}; \mathcal{I} \longmapsto T_P^{\omega}(\mathcal{I})$ of T_P (it has to be distinguished from $T_P{\uparrow}^{\omega}$):

$$T_P^0(\mathcal{I}) = \mathcal{I}, \quad T_P^{n+1}(\mathcal{I}) = T_P(T_P^n(\mathcal{I})) \cup T_P^n(\mathcal{I}), \quad \text{and finally } T_P^{\omega}(\mathcal{I}) = \bigcup_{n=0}^{\infty} T_P^n(\mathcal{I}).$$

These can be used to select a unique Herbrand model among all minimal Herbrand models of $comp(P)$ for any stratified Program P ([ABW88]):

Definition 3.7 ($\mathbf{M}_P^{supp}$)
For $P = P_1 \cup \ldots \cup P_n$ a stratification of P, we define M_P^{supp} as follows:

$$\begin{array}{lcl} M_1 & = & T^{\omega}_{P_1}(\emptyset) \\ M_2 & = & T^{\omega}_{P_2}(M_1) \\ & \ldots & \\ M_n & = & T^{\omega}_{P_n}(M_{n-1}) \\ M_P^{supp} & = & M_n \end{array}$$

To prove *Cumulativity*, we need the following lemma whose simple proof is left to the reader:

Lemma 3.8
Let $U_1 \subseteq U_2$ be (possibly infinite) sets of ground atoms.

a) *If $P \cup U_1 = P_1 \cup \ldots \cup P_n$ is a stratification of $P \cup U_1$, then $P \cup U_2$ is also stratified. A stratification is $P \cup U_2 = P'_1 \cup \ldots \cup P'_n$, where P'_i is obtained from P_i by adding those atoms of U_2, whose relation-symbols already occur in the head of a clause in P_i.*

In addition, we have:

$$\mathcal{I} \subseteq \mathcal{I}' \quad \textit{implies} \quad T^{\omega}_{P_i}(\mathcal{I}) \subseteq T^{\omega}_{P'_i}(\mathcal{I}')$$

b) *If $M_{P\cup U_1}^{supp} \models U_2$ we have:*

$$M'_{i+1} = T^{\omega}_{P'_{i+1}}(M'_i) \ \subseteq \ T^{\omega}_{P_{i+1}}(M_i) = M_{i+1}$$

Corollary 3.9 (Cumulativity and Rationality of M_P^{supp})
The stratified semantics M_P^{supp} is cumulative. As a complete semantics, it is also rational.

Proof: Using a) of the above lemma we get *Cautious Monotony*, while b) proves the *Cut*: this exactly parallels the proof of Theorem 3.4 and is therefore omitted. Because of the completeness (for any atom a: either $M_P^{supp} \models a$ or $M_P^{supp} \models \neg a$) we have: "not $a \mid\!\sim \neg b$" implies "$a \mid\!\sim b$" and *Rationality* is reduced to *Cautious Monotony*. ∎

The main advantage of the supported semantics is that there is a general agreement that this two-valued Herbrand model is the only possible semantics (from the NMR-viewpoint) for stratified programs.

4 WFS and extensions

The drawback of the supported semantics is that it is not defined for the class of *all* programs. Therefore we should look for extensions of the supported semantics defined for the whole class of normal programs. Indeed there are non-stratified programs with a unique intended semantics. We discuss such programs in the second article, where we use them to motivate the stable and the well-founded semantics.

In this section we first recall Przymusinski's definition of the well-founded semantics and use it to prove its *Rationality* (Section 4.1). We then define in Sections 4.2, 4.3 two rational extensions of WFS and show that the regular semantics REG-SEM of You and Yuan satisfies the *Cut* but not *Cautious Monotony*. Section 4.4 presents a semantics that is cumulative but not rational: Pereira's O-SEM. Finally, in Section 4.5 we discuss the relationship of all the semantics considered so far.

4.1 WFS

Here we just give the necessary notions to define WFS and to prove its *Rationality*. A more detailed discussion (showing the close relationship of WFS and STABLE) is contained in the second article. The well-founded model WFS(P) is a 3-valued one defined using the following operator

$$\Phi_P^{\mathcal{J}} : \mathbf{3}_t{}^{B_P} \longmapsto \mathbf{3}_t{}^{B_P};\ \mathcal{I} \longmapsto \Phi_P^{\mathcal{J}}(\mathcal{I})$$

$$\Phi_P^{\mathcal{J}}(\mathcal{I})(A) = \begin{cases} \mathbf{t}, \text{ if there is a clause } A \leftarrow L_1, \ldots, L_n \text{ in } P_{inst} \text{ with:} \\ \quad - \ \forall i \leq n \text{ we have:} \\ \qquad * \ \mathcal{J}(L_i) = \mathbf{t}, \\ \qquad\quad \underline{\text{or}} \\ \qquad * \ L_i \text{ is positive, } \mathcal{J}(L_i) \neq \mathbf{f} \text{ and } \mathcal{I}(L_i) = \mathbf{t}. \\ \mathbf{f}, \text{ if for all clauses } A \leftarrow L_1, \ldots, L_n \in P_{inst} \text{ we have:} \\ \quad - \ \exists i \leq n \text{ with:} \\ \qquad * \ \mathcal{J}(L_i) = \mathbf{f} \\ \qquad\quad \underline{\text{or}} \\ \qquad * \ L_i \text{ is positive, } \mathcal{J}(L_i) \neq \mathbf{t} \text{ and } \mathcal{I}(L_i) = \mathbf{f}. \\ \mathbf{u}, \text{ otherwise.} \end{cases}$$

$\mathcal{J}$ represents the *safe* knowledge: True($\mathcal{J}$) and False($\mathcal{J}$) never decrease. $\mathcal{I}$ stands for the current knowledge; applying $\Phi_P^{\mathcal{J}}$ to $\mathcal{I}$ and iterating this further gives us new information (False($\mathcal{I}$) decreases).

Przymusinski proved that for all three-valued interpretations $\mathcal{J}$, the operator $\Phi_P^{\mathcal{J}}$ is $\leq_t$-*monotone* on $\mathbf{3}_t{}^{B_P}$ and has a (unique) $\leq_t$-*least fixpoint*:

$$lfp(\Phi_P^{\mathcal{J}}) = \Phi_P^{\mathcal{J}}\!\uparrow^{\omega}, \text{ where } \Phi_P^{\mathcal{J}}\!\uparrow^{0} = \langle \emptyset, B_P \rangle.$$

Note that $\Phi_P^{\mathcal{J}}$ is also $\leq_k$-monotone. He then defined the $\leq_k$-monotone operator Ω_P

$$\Omega_P : \mathbf{3}_k{}^{B_P} \longmapsto \mathbf{3}_k{}^{B_P}; \mathcal{J} \longmapsto \Phi_P^{\mathcal{J}}\!\uparrow^{\omega},$$

which has a (unique) $\leq_k$-least fixpoint. This is exactly the well-founded model from *van Gelder, Ross and Schlipf*:

$$\text{WFS}(P) = lfp(\Omega_P).$$

This least fixpoint is a consistent three-valued model, because it is the iteration of a consistency preserving operator. We first prove the following

Lemma 4.1
Let $U_1 \subseteq U_2$ be (possibly infinite) sets of ground atoms.

a) *For $U_2 \cap$ False(WFS($P \cup U_1$)) $= \emptyset$, we have:*

$$\mathcal{J} \leq_k \mathcal{J}' \text{ implies } \Phi_{P \cup U_1}^{\mathcal{J}}\!\uparrow^{\omega} \leq_k \Phi_{P \cup U_2}^{\mathcal{J}'}\!\uparrow^{\omega}.$$

b) *For WFS($P \cup U_1$)$\models U_2$, we have:*

$$\mathcal{J} \leq_k \textit{WFS}(P \cup U_1) \;\; \textit{implies} \;\; \Phi^{\mathcal{J}}_{P \cup U_2}{\uparrow}^{\omega} \leq_k \textit{WFS}(P \cup U_1).$$

Proof: It is easy to see that the following is true for ground atoms A:

- $A \in$ True($\Phi^{\mathcal{J}}_{P \cup U}{\uparrow}^{\omega}$) **iff** there exists a finite $U_0 \subseteq U$: $A \in$ True($\Phi^{\mathcal{J}}_{P \cup U_0}{\uparrow}^{\omega}$),
- $A \in$ False($\Phi^{\mathcal{J}}_{P \cup U}{\uparrow}^{\omega}$) **iff** for all finite $U_0 \subseteq U$: $A \in$ False($\Phi^{\mathcal{J}}_{P \cup U_0}{\uparrow}^{\omega}$).

So we can assume, w.l.o.g. that U_1, U_2 are both finite.

a) Let $\mathcal{J} \leq_k \mathcal{J}'$. It suffices to prove 1. and 2. below:

1. For all $n \in \mathit{IN}$: True($\Phi^{\mathcal{J}}_{P \cup U_1}{\uparrow}^{n}$) $\subseteq$ True($\Phi^{\mathcal{J}'}_{P \cup U_2}{\uparrow}^{n}$). This follows immediately from the definition of $\Phi^{\mathcal{J}}_{P}$ and the assumption $\mathcal{J} \leq_k \mathcal{J}'$.
2. False($\Phi^{\mathcal{J}}_{P \cup U_1}{\uparrow}^{\omega}$) $\subseteq$ False($\Phi^{\mathcal{J}'}_{P \cup U_2}{\uparrow}^{\omega}$).

   This is true, because the assumption in a) ($U_2 \cap$ False(WFS($P \cup U_1$)) $= \emptyset$) and the finiteness of U_2 imply the existence of a $n_0 \in \mathit{IN}$ such that:
   $U_2 \cap$ False($\Phi^{\mathcal{J}}_{P \cup U_1}{\uparrow}^{n_0}$) $= \emptyset$. We then obviously have

   $$\text{False}(\Phi^{\mathcal{J}}_{P \cup U_1}{\uparrow}^{n_0}) \subseteq \text{False}(\Phi^{\mathcal{J}'}_{P \cup U_2}{\uparrow}^{0})$$

   and one shows easily, by inspecting the definition of $\Phi^{\mathcal{J}}_{P}$, that $\Phi^{\mathcal{J}}_{P \cup U_1}$ can be applied on the left-hand side and $\Phi^{\mathcal{J}'}_{P \cup U_2}$can be applied on the right-hand side without changing the subset-inclusion:

   $$\text{for all } i \in \mathit{IN}\text{:} \;\; \text{False}(\Phi^{\mathcal{J}}_{P \cup U_1}{\uparrow}^{n_0+i}) \subseteq \text{False}(\Phi^{\mathcal{J}'}_{P \cup U_2}{\uparrow}^{i}).$$

   This implies the desired result.

b) Obviously, the definition of $\Phi^{\mathcal{J}}_{P}$ implies

$$\Phi^{\mathcal{J}'}_{P \cup U_1}{\uparrow}^{n} = \Phi^{\mathcal{J}'}_{P \cup U_2}{\uparrow}^{n}$$

for all $\mathcal{J}'$ with $U_2 \subseteq$ True($\mathcal{J}'$). Therefore

$$\Phi^{WFS(P \cup U_1)}_{P \cup U_2}{\uparrow}^{\omega} = \Phi^{WFS(P \cup U_1)}_{P \cup U_1}{\uparrow}^{\omega}.$$

Now let $\mathcal{J} \leq_k$ WFS($P \cup U_1$). By $\leq_k$-monotony of $\Omega_P : \mathbf{3}_k{}^{B_P} \longmapsto \mathbf{3}_k{}^{B_P}; \mathcal{I} \longmapsto \Phi^{\mathcal{I}}_{P}\uparrow^{\omega}$

$$\Phi^{\mathcal{J}}_{P \cup U_2}{\uparrow}^{\omega} \leq_k \Phi^{WFS(P \cup U_1)}_{P \cup U_2}{\uparrow}^{\omega} = \Phi^{WFS(P \cup U_1)}_{P \cup U_1}{\uparrow}^{\omega} = \text{WFS}(P \cup U_1).$$

■

Theorem 4.2 (Rationality and Cumulativity of WFS)
WFS is cumulative and rational.

Proof: This is just the application of the previous lemma: use a) to prove *Rationality* and b) to prove *Cut* (similar to the proof of Theorem 3.4).

■

An example of a property being not satisfied by WFS is the *Loop* (see Section 2.4):[7]

Example 4.3 (WFS does not satisfy Loop)

$$\begin{array}{llll} P_{Loop}: & a_1 & \leftarrow & a_0, \neg a_2, \neg a_3 \\ & a_2 & \leftarrow & a_1, \neg a_3, \neg a_0 \\ & a_3 & \leftarrow & a_2, \neg a_0, \neg a_1 \\ & a_0 & \leftarrow & a_3, \neg a_2, \neg a_1 \end{array}$$

We have: WFS$(P_{Loop} \cup \{a_0\}) = \{\neg a_2, \neg a_3, a_1\}$, *WFS*$(P_{Loop} \cup \{a_1\}) = \{\neg a_0, \neg a_3, a_2\}$, *WFS*$(P_{Loop} \cup \{a_2\}) = \{\neg a_1, \neg a_0, a_3\}$ *and WFS*$(P_{Loop} \cup \{a_3\}) = \{\neg a_2, \neg a_1, a_0\}$.

4.2 WFS$^+$

The following example is representative for the problems with *reasoning by cases* (also called *floating conclusions* in [BMS91]) which we can illustrate as follows. Considering only two-valued models, some researchers argue that p should be derivable in the following example and therefore try to define extensions of WFS that behave accordingly on programs of this kind:

Example 4.4 (Case-splitting I)

$$\begin{array}{llll} P_{splitting\,I}: & a & \leftarrow & \neg b \\ & b & \leftarrow & \neg a \\ & p & \leftarrow & a \\ & p & \leftarrow & b \end{array}$$

Although neither a, nor b can be "derived" in any semantics based on two-valued models (as STABLE for example) the disjunction $a \vee b$, thus also p, is true. In this way the example is handled by the completion-semantics too. Baral, Lobo and Minker argue ([BLM90]) that p should be derivable. WFS(P) does not fulfill this.

A straightforward solution to this problem would be to use classical reasoning and just add all consequences true in all minimal models. Let us define:

Definition 4.5 (EWFS, T($\mathcal{J}$), F($\mathcal{J}$))
For a three-valued interpretation $\mathcal{J}$, let $\mathcal{J}$−MIN-MOD(P) denote the class of all minimal two-valued Herbrand models of P that are consistent with $\mathcal{J}$. Furthermore, let

- **T**$(\mathcal{J})$:= *True(* $\mathcal{J}$*−MIN-MOD(P)) and* **F**$(\mathcal{J})$:= *False(* $\mathcal{J}$*−MIN-MOD(P)).*

Here, for a set $\mathcal{S}$ of Herbrand models, True($\mathcal{S}$) (resp. False($\mathcal{S}$)) stands for the set of all ground atoms A, which are true (resp. the negations of which are true) in all models in $\mathcal{S}$[8].

We define

$$EWFS(P) := WFS(P) + \langle \mathbf{T}(WFS(P)), \mathbf{F}(WFS(P)) \rangle.$$

[7]This was noted independently by Cees Witteveen.

[8]If $\mathcal{S} = \emptyset$, we define True($\mathcal{S}$) (resp. False($\mathcal{S}$)) to be equal to $\emptyset$.

EWFS is stronger than WFS (WFS $\leq_k$ EWFS). That it does not satisfy *Cut* can be seen with the following counterexample:

$$
\begin{array}{llll}
P: & a & \leftarrow & \neg a \\
 & \beta & \leftarrow & \neg x, a \\
 & y & \leftarrow & \neg \beta \\
 & z & \leftarrow & \neg y
\end{array}
$$

WFS(P)=$\{\neg x\}$ and therefore EWFS(P)=$\{a,\, \beta,\, \neg x\}$ but WFS($P \cup \{\beta\}$)=$\{\neg x, \neg y, z, \beta\}$ and therefore EWFS$(P \cup \{\beta\}) = \{a,\, \beta,\, \neg x,\, \neg y,\, z\}$. Thus, by adding a derivable atom ("β"), we are able to derive more atoms than without: *Cut* is not satisfied.

We could also iterate the above construction to assure that *Cut* is satisfied: this gives us Baral, Lobo and Minker's semantics GWFS (indeed, this is exactly their definition) which we discuss in the second paper. There we will also see that GWFS has a very strange behaviour (equivalent programs are assigned different semantics): the reason is that *too much literals* are added. There is, however, also the possibility to add only *positive* literals true in all models (after the computation of WFS) and iterating this further until no new literals can be derived. This yields a semantics WFS$^+$ that extends WFS but still is very close to it. The definition of WFS$^+$ stated in [Dix92a] is very technical. Due to some stimulating discussions with H. Herre and J. Schlipf, I found a much more elegant definition. To motivate WFS$^+$ let us consider the following problem suggested by H. Herre. *Given a semantics SEM that does not satisfy Cut, how can we extend SEM to a semantics SEM$^+$ satisfying Cut?* If SEM is *rational* the answer is given by the following

Theorem 4.6 (SEM$^+$)
Let SEM be a rational semantics and define an increasing sequence of sets of atoms M_α by $M_0 := \emptyset$, $M_{\alpha+1} := \{a :\ a \text{ an atom with } a \in SEM(P \cup M_\alpha)\}$ for successor ordinals and $M_\lambda := \bigcup_{\alpha \prec \lambda} M_\alpha$. We have for $\alpha \prec \beta$:

$$SEM(P \cup M_\alpha) \leq_k SEM(P \cup M_\beta).$$

There exists therefore a γ such that $M_\gamma = M_{\gamma+1}$. We define

$$SEM^+(P) := SEM(P \cup M_\gamma).$$

SEM$^+$ is the weakest extension of SEM satisfying Cut. *In addition, SEM$^+$ is also rational and it satisfies the following* closure condition *which is a special instance of the extended Cut: SEM(P+SEM(P))=SEM(P).*

The proof is obvious: we only add those atoms that have to be added in order to satisfy *Cut*. We need *Rationality* to prove inductively SEM$(P \cup M_\alpha) \leq_k$ SEM$(P \cup M_{\alpha+1})$[9] and to reach a limit M_γ. Otherwise the sequence M_α could be oscillating.

Remark 4.7 (Notational Variant of SEM$^+$)
In the above definition of SEM$^+$ we could as well define

$$M_{\alpha+1} := SEM(P \cup M_\alpha)$$

using our convention about treating negative literals in a program (see Remark 2.7). This is justified because both the sequence M_α and the sequence SEM$(P \cup M_\alpha)$ are increasing: if **f** *is associated to a literal, this truthvalue remains and will not be changed.*

[9] The definition of $M_{\alpha+1}$ ensures that if $x \in M_{\alpha+1}$ then $\neg x \notin$ SEM$(P \cup M_\alpha)$ so that *Rationality* can be applied.

The semantics WFS^+ can be seen as an attempt to construct an extension of WFS that satisfies the property of *Supraclassicality* (called *classical completeness* in [Sch92]):

$$\textit{Supraclassicality}\text{: if } P \models a \text{ then } a \in \text{WFS}^+(P) \quad \text{(for atoms } a\text{)}.$$

The technical and much more complicated definition of WFS_C defined in [Sch92] was motivated for the same reasons. Indeed we will show in the second paper that WFS_C is identical with WFS^+ and therefore also shares its abstract properties. A similar result holds for the stable-by-case-semantics introduced in [Sch92] (see the second article).

Definition 4.8 (WFS^+)
Let P be a normal program, and define

$$SEM(P) := WFS(P) \cup \{a : \ a \text{ an atom with: } P \models a\}.$$

The assumptions of Theorem 4.6 are satisfied and we get a cumulative and rational semantics WFS^+:=SEM^+.

To see that the assumptions of the lemma really apply, we only need (besides the rationality of WFS proved in Theorem 4.2) the following observation:

$$P \models a \ \text{ implies } \ \neg a \notin \text{WFS}(P). \quad (*)$$

The reason is that although WFS(P) is in general a three-valued model of P, it can always be extended to a two-valued one: just declare all atoms a with neither $a \in \text{WFS}(P)$ nor $\neg a \in \text{WFS}(P)$ to be true. The resulting two-valued model is obviously a model of P, since in the head of the rules only positive atoms occur.

Let us formally prove that SEM(P) as defined above is rational. Let U be a set of atoms and $U \cap \text{False(SEM(P))} = \emptyset$. We have to show that $\text{SEM}(P) \leq_k \text{SEM}(P \cup U)$. Our assumption implies in particular $U \cap \text{False(WFS(P))} = \emptyset$ and therefore (using rationality of WFS)

$$\text{WFS}(P) \leq_k \text{WFS}(P \cup U).$$

Using the above observation we immediately get

$$\text{SEM}(P) = \text{WFS}(P) + \{a : \ P \models a\} \leq_k \text{WFS}(P \cup U) = \{a : \ P \cup U \models a\} = \text{SEM}(P \cup U).$$

Theorem 4.9 (Characterization of WFS^+)
WFS^+ is the weakest extension of WFS that satisfies Cut and Supraclassicality. In addition WFS^+ is rational, satisfies the closure condition and the following property (stronger than Supraclassicality):

$$\textit{Generalized Supraclassicality: If } P \cup WFS^+(P) \models a \textit{ then } a \in WFS^+(P) \quad (a \textit{ an atom}).$$

Again the proof is simple: *Supraclassicality* holds by construction and only those atoms are added that need to be added in order to satisfy *Cut*. We get the stronger version of *Supraclassicality* by free: this is because of the inductive definition using Remark 4.7.

As already mentioned after Remark 2.7, WFS^+ does not satisfy the *extended Cut*. The deeper reason for this is, as we will see in the second article, the failure of the *Unfolding* property.

4.3 WFS$'$ and REG-SEM

WFS$^+$ derives a from programs containing "$a \leftarrow \neg a$". A more sceptical variant where such literals are not derivable (they are assigned **u**) can be obtained by considering not all (minimal) models extending WFS, but only models $\mathcal{A}'$ that satisfy certain conditions. To be precise, let us formulate a condition that *excludes* models assigning a to be "true" in programs containing $a \leftarrow \neg a$:[10]

C$_3$: If all rule bodies for a are false in $\mathcal{A}$ then $\mathcal{A} \models \neg a$.

It turns out that all three-valued stable models (see [Prz91]) have this property. This class has been recently shown to coincide with the class of *justified* models of You and Yuan defined in [YY90] (see [YY93]):

Definition 4.10 (Justified and Regular Models)
A three-valued Herbrand model $\mathcal{M}$ is justified for a program P, if every $a \in True(\mathcal{M})$ is justified. The notion of justifiedness of an atom a is defined inductively: a is justified if

- *there is a clause $a \leftarrow \neg c_1, \ldots, \neg c_m$ in P_{inst} such that $\{c_1, \ldots, c_m\} \subseteq False(\mathcal{M})$, or*
- *there is a clause $a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m$ in P_{inst} such that $b_1, \ldots, b_n$ are already justified and $\{c_1, \ldots, c_m\} \subseteq False(\mathcal{M})$.*

We define

$$\begin{array}{lll} \textit{JUST-MOD}(P) & := \{\mathcal{M} : \mathcal{M} \in \textit{MOD}_{3-val}^{Herb_{\mathcal{L}_P}}(P) \text{ and } \mathcal{M} \text{ is justified for } P\} \\ \textit{REG-MOD}(P) & := \{\mathcal{M} : \mathcal{M} \text{ is } \leq_k\text{-maximal in JUST-MOD(P)}\} \end{array}$$

The class REG-MOD(P) of regular models of a program P was proposed in [YY90] as a semantics for logic programs. In [YY93] they showed that this class coincides with the class of *partial stable models* of Sacca and Zaniolo ([SZ90]) as well as with Dung's *preferred extensions* ([Dun91]) as well as with Przymusinski's $\leq_k$-maximal three-valued stable models ([Prz91]). In particular all the induced entailment relations coincide. We use the notation REG-SEM for the sceptical consequence relation induced by the set of all regular models

$$\text{REG-SEM}(P) := \bigcap_{\mathcal{M} \in \text{REG-MOD}(P)} \{L : L \text{ is a pos. or neg. literal with: } \mathcal{M} \models L\}$$

Przymusinski proved in [Prz90] that the induced consequence relation of all three-valued stable models coincides with WFS. Since the regular models are $\leq_k$-maximal three-valued stable models, they are $\leq_k$-extensions of the wellfounded model. Unfortunately, REG-SEM is not cumulative (the following program P_{You} was found during an e-mail discussion with Jia You):

Example 4.11 (REG-SEM is not cumulative)

$$\begin{array}{llll} P_{You}: & a & \leftarrow & \neg b \\ & c & \leftarrow & \neg b \\ & b & \leftarrow & c, \neg a \end{array} \qquad \begin{array}{llll} P_{\text{REG/WFS}^+}: & a & \leftarrow & \neg b \\ & c & \leftarrow & \neg b \\ & b & \leftarrow & c, \neg a \\ & p & \leftarrow & \neg p \end{array}$$

[10] The notation C$_3$ is chosen to be in accordance with [Dix92a] and the second article [Dix94].

P_{You} has exactly one regular model: $\{a, \neg b, c\}$ which coincides with the unique stable model. Therefore a and c are the only derivable literals. But if we add c to P we get an additional regular model: $\{\neg a, b, c\}$. Thus a does no longer hold.

For the second program, REG-SEM only derives a and c but not p or b (nor $\neg p$ or $\neg b$). Since WFS$^+(P_{\text{REG/WFS}^+}) = \{p\}$, both semantics are incomparable with respect to $\leq_k$.

We are now ready to define a restricted version WFS$'$ of WFS$^+$ by requiring that the added atoms are *both* contained in all classical models and in all regular models. We therefore define:

Definition 4.12 (WFS$'$)
Let P be a normal program, and define

$$SEM(P) := WFS(P) \cup \{a : \ a \text{ an atom with } P \models a \text{ and } REG\text{-}MOD(P) \models a\}.$$

The assumptions of Theorem 4.6 are satisfied: we get a rational semantics WFS$'$:=SEM$^+$.

WFS$'$ is a rational and cumulative extension of WFS:

Theorem 4.13 (Rationality of WFS$'$)
WFS$'$ is a rational extension of WFS that satisfies Cut and C_3, i.e. a cumulative and rational extension of WFS. Moreover, we have: WFS $\leq_k$ WFS$'$ $\leq_k$ WFS$^+$.

From the definition of WFS$'$ it is obvious that REG-SEM is a $\leq_k$-extension of WFS$'$. The next lemma (the proof of which is obvious) will be used to prove the *Cut*:

Lemma 4.14
If $\mathcal{A}$ is a justified model for P then there is a regular model $\mathcal{A}'$ of P with $\mathcal{A} \leq_k \mathcal{A}'$.

Lemma 4.15 (Cut holds for REG-SEM)
Let $U_1 \subseteq U_2$ be (infinite) sets of ground atoms. In addition, let REG-SEM($P \cup U_1$)$\models U_2$. Then: Every regular model of $P \cup U_2$ is also a regular model of $P \cup U_1$.

Proof: The assumption implies that every regular model $\mathcal{M}$ of $P \cup U_1$ is also a model of U_2 and hence a justifiable model for $P \cup U_2$. Now $\mathcal{M}$ can be extended (previous lemma) to a regular model $\mathcal{M}'$ for $P \cup U_2$. Since $\mathcal{M}'$ still is justifiable for $P \cup U_1$ we have $\mathcal{M}' = \mathcal{M}'$ ($\mathcal{M}$ is $\leq_k$-maximal). ∎

In fact, REG-SEM also satisfies the *extended Cut* as we will show in the second article. This is because the general *unfolding* property holds for REG-SEM and this property implies automatically (with two other weak properties) the extended Cut.

4.4 O-SEM

Recently, Pereira et al. ([PAA92]) defined an extension of WFS, called O-SEM, by adding suitable negative literals. Let us consider the following two programs

Example 4.16 (O-SEM)

$$P_{O-SEM1}: \begin{array}{rcl} a & \leftarrow & \neg a \\ c & \leftarrow & \neg a \end{array} \qquad P_{O-SEM2}: \begin{array}{rcl} a & \leftarrow & \neg a \\ p & \leftarrow & a \\ q & \leftarrow & \neg p \end{array}$$

Pereira et al. argue that the semantics of the first program should be $\{\neg c\}$. *They also derive* $\{\neg p, q\}$ *from* $P_{O-SEM\,2}$ *which shows that their semantics is not rational:* $\neg a$ *is not derivable from* $P_{O-SEM\,2}$, $\neg p$ *is derivable from* $P_{O-SEM\,2}$ *but yet* p *is derivable from* $P_{O-SEM\,2} \cup \{a\}$ *(and therefore* $\neg p$ *is not derivable, as Rationality would require).*

Their semantics also does not satisfy the condition (termed C_4 in [PAA92])

"If $\mathcal{A} \models \neg a$ then all rule bodies for a are false in $\mathcal{A}$."

and, indeed, is not even a 3-valued model of P: c is false but all rule bodies of c are undefined. Strictly speaking, this shows that O-SEM does not fit into our framework of Definition 2.4. But we can easily incorporate O-SEM by weakening the truth-table of "$\leftarrow$": we simply evaluate "$\mathbf{f} \leftarrow \mathbf{u}$" to $\mathbf{t}$. This does not affect our use of three-valued logic in the preceding sections.

However, we can prove that this semantics is cumulative: it is therefore the only known cumulative but not rational semantics for normal programs.

Theorem 4.17 (Cumulativity of O-SEM)
Pereira's semantics O-SEM is cumulative but not rational.

4.5 The relationship between the semantics

It is obvious that WFS and COMP are incomparable with respect to $\leq_k$, since already simple loops are handled differently. For the program P consisting of $p \leftarrow p$, we have

$$\text{COMP}(P) = \{l : \ l \in Th(\{p \leftrightarrow p\})\} = \emptyset \not\leq_k \{\neg p\} = \text{WFS}(P)$$

while for P_{comp} (see Example 3.2) we get

$$\text{WFS}(P_{comp}) = \{\neg b\} \not\leq_k \{a, b, p\} = \{l : \ l \in comp(P_{comp})\} = \text{COMP}(P_{comp}).$$

It is also obvious that $\text{COMP}_3 \leq_k$ WFS and $\text{COMP}_3 \leq_k$COMP since WFS(P) is a three-valued model of P and COMP_3 is the $\leq_k$-least such model.

The relations between WFS, WFS$'$ and WFS$^+$ are trivial by construction. An easy inspection of O-SEM shows the incomparability of WFS$'$ and O-SEM: O-SEM($P_{splitting\ I}$)=$\emptyset$ while WFS$'$($P_{O-SEM\,1}$)=$\emptyset$.

Let us again note that REG-SEM, the semantics based on regular models ([YY90]) is representative for the following semantics (proved to be equivalent by You and Yuan ([YY93])

- Sacca and Zaniolo's *partial models* ([SZ90]),
- Przymusinski's $\leq_k$*-maximal 3-valued stable models* ([Prz91, Prz90]), and
- Dung's *preferred extensions* ([Dun91]).

.

All these relations are illustrated in Figure 3.

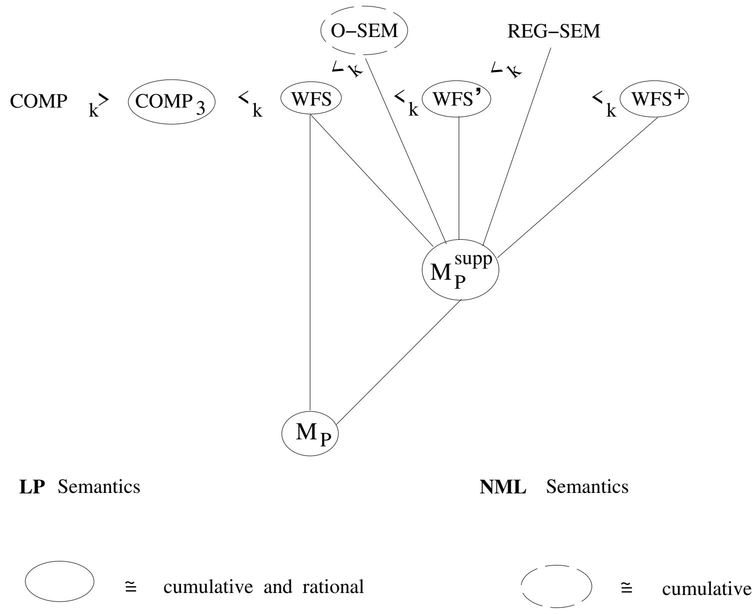

Figure 3: Semantics for Normal Programs

5 Concluding remarks

The fact that there has been defined an enormous number of different semantics for logic programs in recent years, reflects very clearly the situation in this area: there is still no general agreement about the *right* semantics. We argue that this question should be answered by more general principles and not just by examples and counterexamples appealing more or less to one's intuitions.

Our approach consists of associating to any semantics a sceptical non-monotonic entailment relation and investigating its (non-monotonic) properties.

While we concentrated in this article on *strong* properties induced by Kraus, Lehmann and Magidor's work in general non-monotonic reasoning, we will supplement these principles with a set of *weak* properties in the forthcoming second article of this series ([Dix94]).

Acknowledgements

Many thanks to Martin Müller who proofread this paper and made a lot of helpful remarks during many talks. Discussions with Heinrich Herre and useful suggestions of John Schlipf also helped to improve this paper a lot. Roland Bol suggested to consider $\leq_k$-maximal stable models and Michael Gelfond turned the author's attention to You and Yuan's work. E-mail discussions with Li-Yan Yuan and Jia-Huai You helped to correct some hasty conjectures. In addition, Li-Yan Yuan made some important observations about the *extended* versions of the abstract rules (see Remark 2.7) which resulted in a careful reformulation of some results.

Last but not least an anonymous referee read the paper extremely carefully and still found lots of inaccuracies. I hope to have cleaned up most of them.

References

[AB94] K.R. Apt and R. Bol. Logic Programming and Negation: A Survey. *Journal of Logic Programming*, 19-20:9–71, 1994.

[ABW88] K. Apt, H. Blair, and A. Walker. Towards a theory of declarative knowledge. In Jack Minker, editor, *Foundations of Deductive Databases*, chapter 2, pages 89–148. Morgan Kaufmann, 1988.

[Apt90] K. R. Apt. Logic programming. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Vol. B*, chapter 10, pages 493–574. Elsevier Science Publishers, 1990.

[BF91] Nicole Bidoit and Christine Froidevaux. Negation by Default and unstratifiable logic Programs. *Theoretical Computer Science*, 78:85–112, 1991.

[BLM90] Chitta Baral, Jorge Lobo, and Jack Minker. Generalized Well-founded Semantics for Logic Programs. In M. E. Stickel, editor, *10th International Conference on Automated Deduction, LNAI 449, subseries LNCS*, pages 102–116. Springer, J. Siekmann, July 1990.

[BMS91] Gerhard Brewka, David Makinson, and Karl Schlechta. A Cumulative Inference Relation for JTMS and Logic Programming. In J. Dix, K. P. Jantke, and P. H. Schmitt, editors, *Proceedings of the First Workshop on Nonmonotonic and Inductive Logic, Karlsruhe, 1990*, pages 1–12. LNCS 543, Springer, October 1991.

[Bol90] Roland N. Bol. Loop Checking and Negation. In J. van Eijck, editor, *Logics in AI*, pages 121–138. Springer, Lecture Notes in Artificial Intelligence 478, September 1990.

[Bol91] Roland N. Bol. *Loop Checking in Logic Programming*. PhD thesis, University of Amsterdam, 1991.

[Bol93] Roland Bol. Loop checking and Negation. *Journal of Logic Programming*, 15(2):147–175, 1993.

[CK91] Jianhua Chen and Sukhamay Kundu. The Strong Semantics For Logic Programs. In Z.W. Ras and M. Zemankova, editors, *Proceedings of the 6th Int. Symp. on Methodologies for Intelligent Systems, Charlotte, NC, 1991*, pages 490–499. Springer, Lecture Notes in Artificial Intelligence 542, 1991.

[CKPR73] A. Colmerauer, H. Kanoui, R. Pasero, and P. Roussel. Un systeme de communication homme-machine en français. Technical report, Groupe de Intelligence Artificielle Universite de Aix-Marseille II, 1973.

[CL89] L. Cavedon and J.W. Lloyd. A Completeness Theorem for SLDNF-Resolution. *Journal of Logic Programming*, 7:177–191, 1989.

[Cla78] K. L. Clark. Negation as Failure. In H. Gallaire and J. Minker, editors, *Logic and Data-Bases*, pages 293–322. Plenum, New York, 78.

[DC90] Hendrik Decker and Lawrence Cavedon. Generalizing syntactic properties which ensure that SLDNF-Resolution is complete and flounder-free. Technical report, ECRC Munich, January 1990.

[Dix91a] Jürgen Dix. Classifying Semantics of Logic Programs. In Anil Nerode, Wiktor Marek, and V. S. Subrahmanian, editors, *Logic Programming and Non-Monotonic Reasoning, Proceedings of the first International Workshop*, pages 166–180. Washington D.C, MIT Press, July 1991.

[Dix91b] Jürgen Dix. Cumulativity and Rationality in Semantics of Normal Logic Programs. In J. Dix, K. P. Jantke, and P. H. Schmitt, editors, *Proceedings of the first Workshop on Nonmonotonic and Inductive Logic 1990 in Karlsruhe*, pages 13–37. LNCS 543, Springer, October 1991.

[Dix92a] Jürgen Dix. A Framework for Representing and Characterizing Semantics of Logic Programs. In B. Nebel, C. Rich, and W. Swartout, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR '92)*, pages 591–602. San Mateo, CA, Morgan Kaufmann, 1992.

[Dix92b] Jürgen Dix. Classifying Semantics of Disjunctive Logic Programs. In K. Apt, editor, *LOGIC PROGRAMMING: Proceedings of the 1992 Joint International Conference and Symposium*, pages 798–812. MIT Press, November 1992.

[Dix92c] Jürgen Dix. *Nichtmonotones Schließen und dessen Anwendung auf Semantiken logischer Programme*. PhD thesis, University of Karlsruhe, 1992.

[Dix94] Jürgen Dix. A Classification-Theory of Semantics of Normal Logic Programs: II. Weak Properties. *Fundamenta Informaticae*, forthcoming, 1994.

[Dix95] Jürgen Dix. Semantics of Logic Programs: Their Intuitions and Formal Properties. An Overview. In Andre Fuhrmann and Hans Rott, editors, *Logic, Action and Information. Proceedings of the Konstanz Colloquium in Logic and Information (LogIn '92)*, pages 227–313. DeGruyter, 1995.

[DM92] Jürgen Dix and David Makinson. The Relationship between KLM and MAK models for Nonmonotonic Inference Operations. *Journal of Logic, Language and Information*, Volume 1, No. 2:131–140, 1992.

[DM94a] Jürgen Dix and Martin Müller. An Axiomatic Framework for Representing and Characterizing Semantics of Disjunctive Logic Programs. In Pascal Van Hentenryck, editor, *Proceedings of the 11th Int. Conf. on Logic Programming, S. Margherita Ligure*, pages 303–322. MIT, June 1994.

[DM94b] Jürgen Dix and Martin Müller. Partial Evaluation and Relevance for Approximations of the Stable Semantics. In Z.W. Ras and M. Zemankova, editors, *Proceedings of the 8th Int. Symp. on Methodologies for Intelligent Systems, Charlotte, NC, 1994*, pages 511–520. Springer, Lecture Notes in Artificial Intelligence 869, 1994.

[DM94c] Jürgen Dix and Martin Müller. The Stable Semantics and its Variants: A Comparison of Recent Approaches. In L. Dreschler-Fischer and B. Nebel, editors, *Proceedings of the 18th German Annual Conference on Artificial Intelligence (KI '94), Saarbrücken, Germany*, pages 82–93. Springer, Lecture Notes in Artificial Intelligence 861, 1994.

[Dun91] P. M. Dung. Negation as Hypotheses: An Abductive Foundation for Logic Programming. In Koichi Furukawa, editor, *Proceedings of the 8th Int. Conf. on Logic Programming, Paris*. MIT, June 1991.

[Fit85] Melvin Fitting. A Kripke-Kleene Semantics of logic Programs. *Journal of Logic Programming*, 4:295–312, 1985.

[GL88] Michael Gelfond and Vladimir Lifschitz. The Stable Model Semantics for Logic Programming. In R. Kowalski and K. Bowen, editors, *5th Conference on Logic Programming*, pages 1070–1080. MIT Press, 1988.

[HY91] Yong Hu and Li Yan Yuan. Extended Well-Founded Model Semantics for General Logic Programs. In Koichi Furukawa, editor, *Proceedings of the 8th Int. Conf. on Logic Programming, Paris*, pages 412–425. MIT, June 1991.

[KLM90] Sarit Kraus, Daniel Lehmann, and Menachem Magidor. Nonmonotonic Reasoning, Preferential Models and Cumulative Logics. *Artificial Intelligence*, 44(1):167–207, 1990.

[Kow74] R.A. Kowalski. Predicate logic as a programming language. In *Proceeedings IFIP' 74*, pages 569–574. North Holland Publishing Company, 1974.

[Kun87] Kenneth Kunen. Negation in Logic Programming. *Journal of Logic Programming*, 4:289–308, 1987.

[Llo87] John Lloyd. *Foundations of Logic Programming*. Springer- Verlag, 2nd edition, 1987.

[LM92] Daniel Lehmann and Menachem Magidor. What does a conditional knowledge base entail? *Artificial Intelligence*, 55:1–60, 1992.

[LMR92] Jorge Lobo, Jack Minker, and Arcot Rajasekar. *Foundations of Disjunctive Logic Programming*. MIT-Press, 1992.

[Mak89] David Makinson. General Theory of Cumulative Inference. In Reinfrank, de Kleer, Ginsberg, and Sandewall, editors, *Non-Monotonic Reasoning*. Lecture Notes in Artificial Intelligence 346, Springer-Verlag, January 1989.

[Mak94] David Makinson. General Patterns in Nonmonotonic Reasoning. In D.M. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming Vol. 2, Nonmonotonic and Uncertain Reasoning*, chapter 2.2. Oxford University Press, 1994.

[Min88] Jack Minker. *Foundations of Deductive Databases*. Morgan Kaufmann, 95 First Street, Los Altos, CA 94022, 1st edition, 1988.

[Min93] Jack Minker. An Overview of Nonmonotonic Reasoning and Logic Programming. *Journal of Logic Programming, Special Issue*, 17, 1993.

[PAA92] L.M. Pereira, J.J. Alferes, and J.N. Aparicio. Adding Closed World Assumptions to Well Founded Semantics. In ICOT, editor, *Fifth Generation Computer Systems, Proceedings of the Conference*, pages 562–569. OMSHA, June 1992.

[Prz90] Teodor Przymusinski. Well-founded Semantics Coincides With Three-Valued Stable Semantics. *Fundamenta Informaticae*, XIII:445–463, 1990.

[Prz91] T. C. Przymusinski. Three-valued non-monotonic formalisms and semantics of logic programs. *Artificial Intelligence*, 49:309–343, 1991. (Extended abstract appeared in: Three-valued non-monotonic formalisms and logic programming. *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning (KR'89), Toronto, Canada*, pages 341–348, Morgan Kaufmann, 1989.).

[Sch92] John S. Schlipf. Formalizing a Logic for Logic Programming. *Annals of Mathematics and Artificial Intelligence*, 5:279–302, 1992.

[Stä94] Robert F. Stärk. Input/output dependencies of normal logic programs. *Journal of Logic and Computation*, 4(3):249–262, 1994.

[SZ90] D. Sacca and C. Zaniolo. Stable models and non-determinism in logic programs with negation. In *Proc. the 9th ACM PODS*, pages 205–217, 1990.

[SZ91] D. Sacca and C. Zaniolo. Partial models and Three-Valued Models in Logic Programs with Negation. In Anil Nerode, Wiktor Marek, and V. S. Subrahmanian, editors, *Logic Programming and Non-Monotonic Reasoning, Proceedings of the first International Workshop*, pages 87–104. Washington D.C, MIT Press, July 1991.

[vEK76] M.H. van Emden and R.A. Kowalski. The semantics of predicate logic as a programming language. *JACM*, 23:733–742, 1976.

[vGRS88] Allen van Gelder, Kenneth Ross, and J.S. Schlipf. Unfounded Sets and well-founded Semantics for general logic Programs. In *Proceedings 7th Symposion on Principles of Database Systems*, pages 221–230, 1988.

[vGRS91] Allen van Gelder, Kenneth Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38:620–650, 1991.

[YY90] Jia-Huai You and Li-Yan Yuan. Three-valued Formalization of Logic Programming: is it needed. In *Proc. of the 9th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems,*, pages 172–182. ACM Press, 1990.

[YY93] Jia-Huai You and Li-Yan Yuan. On the Equivalence of Semantics for Normal Logic Programs. In *Proceedings of Workshop on Logic Programming with Incomplete Information, Vancouver Oct. 1993, following ILPS' 93*, pages 161–176, 1993. available as Research Report of ISIS-RR-93-E12, Fujitsu Labs.

Contents