Level Mapping Characterizations of Selector-Generated Models for Logic Programs

Sibylle Schwarz

Pascal Hitzler
pascal.hitzler@wright.edu

Follow this and additional works at: http://corescholar.libraries.wright.edu/cse

Part of the Bioinformatics Commons, Communication Technology and New Media Commons, Databases and Information Systems Commons, OS and Networks Commons, and the Science and Technology Studies Commons

Repository Citation
http://corescholar.libraries.wright.edu/cse/63

This Conference Proceeding is brought to you for free and open access by Wright State University's CORE Scholar. It has been accepted for inclusion in Computer Science and Engineering Faculty Publications by an authorized administrator of CORE Scholar. For more information, please contact corescholar@www.libraries.wright.edu.
Level Mapping Characterizations of Selector Generated Models for Logic Programs

Pascal Hitzler\textsuperscript{1} and Sibylle Schwarz\textsuperscript{2}

\textsuperscript{1} AIFB, Universität Karlsruhe (TH)
email: hitzler@aifb.uni-karlsruhe.de

\textsuperscript{2} Institut für Informatik, Martin-Luther-Universität Halle-Wittenberg
email: schwarzs@informatik.uni-halle.de

Abstract. Assigning semantics to logic programs via selector generated models (Schwarz 2002/2003) extends several semantics, like the stable, the inflationary, and the stable generated semantics, to programs with arbitrary formulae in rule heads and bodies. We study this approach by means of a unifying framework for characterizing different logic programming semantics using level mappings (Hitzler and Wendt 200x, Hitzler 2003), thereby supporting the claim that this framework is very flexible and applicable to very diversely defined semantics.

1 Introduction

Hitzler and Wendt [8, 10, 11] have recently proposed a unifying framework for different logic programming semantics. This approach is very flexible and allows to cast semantics of very different origin and style into uniform characterizations using level mappings, i.e. mappings from atoms to ordinals, in the spirit of the definition of acceptable programs [2], the use of stratification [1, 14] and a characterization of stable models by Fages [3]. These characterizations display syntactic and semantic dependencies between language elements by means of the preorders on ground atoms induced by the level mappings, and thus allow inspection of and comparison between different semantics, as exhibited in [8, 10, 11].

For the syntactically restricted class of normal logic programs, the most important semantics — and some others — have already been characterized and compared, and this was spelled out in [8, 10, 11]. Due to the inherent flexibility of the framework, it is clear that studies of extended syntax are also possible, but have so far not been carried out. In this paper, we will present a non-trivial technical result which provides a first step towards a comprehensive comparative study of different semantics for logic programs under extended syntax.

\textsuperscript{*} The first named author acknowledges support by the German Federal Ministry of Education and Research under the SmartWeb project, and by the European Union under the KnowledgeWeb Network of Excellence. He also acknowledges the hospitality of the Graduiertenkolleg \textit{Wissensrepräsentation} at the University of Leipzig, Germany, while working on a first draft of this paper.
Table 1. Notions of specific types of rules.

<table>
<thead>
<tr>
<th>rule is called</th>
<th>set</th>
<th>condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>definite</td>
<td>LP</td>
<td>body(r) ∈ Lg({∧, t}, A) and head(r) ∈ A</td>
</tr>
<tr>
<td>normal</td>
<td>NLP</td>
<td>body(r) ∈ Lg({∧, t}, Lit(A)) and head(r) ∈ A</td>
</tr>
<tr>
<td>head-atomic</td>
<td>HALP</td>
<td>body(r) ∈ Lg({∧}, Lit(A)) and head(r) ∈ A</td>
</tr>
<tr>
<td>pos. head disj.</td>
<td>DLP</td>
<td>body(r) ∈ Lg({∧, t}, Lit(A)) and head(r) ∈ Lg({\lor}, A)</td>
</tr>
<tr>
<td>disjunctive</td>
<td>DLP</td>
<td>body(r) ∈ Lg({∧, t}, Lit(A)), head(r) ∈ Lg({\lor, f}, Lit(A))</td>
</tr>
<tr>
<td>head-disjunctive</td>
<td>HDLP</td>
<td>body(r) ∈ Lg({\lor}, A), head(r) ∈ Lg({\lor, f}, Lit(A))</td>
</tr>
<tr>
<td>generalized</td>
<td>GLP</td>
<td>no condition</td>
</tr>
</tbody>
</table>

More precisely, among the many proposals for semantics for logic programs under extended syntax we will study a very general approach due to Schwarz [15, 16]. In this framework, arbitrary formulae are allowed in rule heads and bodies, and it encompasses the inflationary semantics [12], the stable semantics for normal and disjunctive programs [5, 13], and the stable generated semantics [7]. It can itself be understood as a unifying framework for different semantics.

In this paper, we will provide a single theorem — and some corollaries thereof — which gives a characterization of general selector generated models by means of level mappings. It thus provides a link between these two frameworks, and implicitly yields level mapping characterizations of the semantics encompassed by the selector generated approach.

The plan of the paper is as follows. In Section 2 we will fix preliminaries and notation. In Section 3 we will review selector generated models as introduced in [15, 16]. In Section 4, we present our main result, Theorem 4, which gives a level-mapping characterization of general selector generated models in the style of [8, 10, 11]. In Section 5 we study corollaries from Theorem 4 concerning specific cases of interest encompassed by the result. We eventually conclude and discuss further work in Section 6.

2 Preliminaries

Throughout the paper, we will consider a language \( \mathcal{L} \) of propositional logic over some set of propositional variables, or atoms, \( A \), and connectives \( \Sigma_{cl} = \{\neg, \lor, \land, t, f\} \), as usual. A rule \( r \) is a pair of formulae from \( \mathcal{L} \) denoted by \( \varphi \Rightarrow \psi \). \( \varphi \) is called the body of the rule, denoted by body(r), and \( \psi \) is called the head of the rule, denoted by head(r). A program is a set of rules. A literal is an atom or a negated atom, and Lit(A) denotes the set of all literals in \( \mathcal{L} \). For a set of connectives \( C \subseteq \Sigma_{cl} \) we denote by \( \text{Lg}(C, A) \) the set of all formulae over \( \mathcal{L} \) in which only connectives from \( C \) occur.

Further terminology is introduced in Table 1. The abbreviations in the second column denote the sets of all rules with the corresponding property. A program containing only definite (normal, etc.) rules is called definite (normal, etc.). Programs not containing the negation symbol \( \neg \) are called positive. Facts are rules \( r \) where body(r) = t, denoted by \( \Rightarrow \text{head}(r) \).
The base \( B_P \) is the set of all atoms occurring in a program \( P \). A two-valued interpretation of a program \( P \) is represented by a subset of \( B_P \), as usual. By \( I_P \), we denote the set of all interpretations of \( P \). It is a complete lattice with respect to the subset ordering \( \subseteq \). For an interpretation \( I \in I_P \), we define \( \uparrow I = \{ J \in I_P \mid I \subseteq J \} \) and \( \downarrow I = \{ J \in I_P \mid J \subseteq I \} \). \( [I, J] = \uparrow I \cap \downarrow J \) is called an interval of interpretations.

The model relation \( M \models \varphi \) for an interpretation \( M \) and a propositional formula \( \varphi \) is defined as usual in propositional logic, and \( \text{Mod}(\varphi) \) denotes the set of all models of \( \varphi \). Two formulae \( \varphi \) and \( \psi \) are logically equivalent, written \( \varphi \equiv \psi \), iff \( \text{Mod}(\varphi) = \text{Mod}(\psi) \).

A formula \( \varphi \) is satisfied by a set \( J \subseteq I_P \) of interpretations if each interpretation \( J \in J \) is a model of \( \varphi \). For a program \( P \), a set \( J \subseteq I_P \) of interpretations determines the set of all rules which fire under \( J \), formally \( \text{fire}(P, J) = \{ r \in P \mid \forall J \in J : J \models \text{body}(r) \} \). An interpretation \( M \) is called a model of a rule \( r \) (or satisfies \( r \)) if \( M \) is a model of the formula \( \neg \text{body}(r) \lor \text{head}(r) \). An interpretation \( M \) is a model of a program \( P \) if it satisfies each rule in \( P \).

For conjunctions or disjunctions \( \varphi \) of literals, \( \varphi^+ \) denotes the set of all atoms occurring positively in \( \varphi \), and \( \varphi^- \) contains all atoms that occur negated in \( \varphi \). For instance, for the formula \( \varphi = (a \land \neg b \land \neg a) \) we have \( \varphi^+ = \{ a \} \) and \( \varphi^- = \{ a, b \} \). In heads \( \varphi \) consisting only of disjunctions of literals, we always assume without loss of generality that \( \varphi^+ \cap \varphi^- = \emptyset \).

If \( \varphi \) is a conjunction of literals, we abbreviate \( M \models \bigwedge_{a \in \varphi^+} a \) (i.e. \( \varphi^+ \subseteq M \)) by \( M \models \varphi^+ \) and \( M \models \bigwedge_{a \in \varphi^-} \neg a \) (i.e. \( \varphi^- \cap M = \emptyset \)) by \( M \models \varphi^- \), abusing notation. If \( \varphi \) is a disjunction of literals, we write \( M \models \varphi^+ \) for \( M \models \bigvee_{a \in \varphi^+} a \) (i.e. \( M \cap \varphi^+ \neq \emptyset \)) and \( M \models \varphi^- \) for \( M \models \bigvee_{a \in \varphi^-} \neg a \) (i.e. \( \varphi^- \not\subseteq M \)).

By iterative application of rules from a program \( P \subseteq \text{GLP} \) starting in the least interpretation \( \emptyset \in I_P \), we can create monotonically increasing (transfinite) sequences of interpretations of the program \( P \), as follows.

**Definition 1.** A (transfinite) sequence \( C \) of length \( \alpha \) of interpretations of a program \( P \subseteq \text{GLP} \) is called a \( P \)-chain iff

\[
\text{(C0)} \quad C_0 = \emptyset,
\]

\[
\text{(C}\beta\text{)} \quad C_{\beta+1} \in \text{Min}(\uparrow C_\beta \cap \text{Mod}(\text{head}(Q_\beta))) \text{ for some set of rules } Q_\beta \subseteq P \text{ and for all } \beta \text{ with } \beta + 1 < \alpha, \text{ and}
\]

\[
\text{(C}\lambda\text{)} \quad C_\lambda = \bigcup \{ C_\beta \mid \beta < \lambda \} \text{ for all limit ordinals } \lambda < \alpha.
\]

\( C_P \) denotes the collection of all \( P \)-chains.

Note that all \( P \)-chains increase monotonically with respect to \( \subseteq \).

### 3 Selector generated models

In [15, 16], a framework for defining declarative semantics of generalized logic programs was introduced, which encompasses several other semantics, as already mentioned in the introduction. Parametrization within this framework is done via so-called selector functions, defined as follows.
Definition 2. A selector is a function \( \text{Sel} : \mathbb{C} \times \mathbb{I} \to 2^{\mathbb{I}} \), satisfying \( \emptyset \neq \text{Sel}(C, I) \subseteq [I, \sup(C)] \) for all \( \mathbb{P} \)-chains \( C \) and each interpretation \( I \in \downarrow \sup(C) \).

We use selectors \( \text{Sel} \) to define nondeterministic successor functions \( \Omega \mathbb{P} \) on \( \mathbb{I} \), as follows.

Definition 3. Given a selector \( \text{Sel} : \mathbb{C} \times \mathbb{I} \to 2^{\mathbb{I}} \) and a program \( \mathbb{P} \), the function \( \Omega \mathbb{P} : (\mathbb{C} \times \mathbb{I}) \times (\mathbb{C} \times \mathbb{I}) \to 2^{\mathbb{I}} \) is defined by

\[
\Omega \mathbb{P}(\text{Sel}, C, I) = \min ([I, \sup(C)] \cap \text{Mod}(\text{head}(\text{fire}(\mathbb{P}, \text{Sel}(C, I))))) .
\]

Example 1. In this paper, we will have a closer look at the following selectors.

- lower bound selector \( \text{Sel}_l(C, I) = \{I\} \)
- lower and upper bound selector \( \text{Sel}_{lu}(C, I) = \{I, \sup(C)\} \)
- interval selector \( \text{Sel}_i(C, I) = [I, \sup(C)] \)
- chain selector \( \text{Sel}_c(C, I) = [I, \sup(C)] \cap C \)

With the first two arguments (the selector \( \text{Sel} \) and the chain \( C \)) fixed, the function \( \Omega \mathbb{P}(\text{Sel}, C, I) \) can be understood as a nondeterministic consequence operator. Iteration of the function \( \Omega \mathbb{P}(\text{Sel}, C, \cdot) \) from the least interpretation \( \emptyset \) creates sequences of interpretations. This leads to the following definition of \((\mathbb{P}, \mathbb{M}, \text{Sel})\)-chains.

Definition 4. A \((\mathbb{P}, \mathbb{M}, \text{Sel})\)-chain is a \( \mathbb{P} \)-chain satisfying \( (C_{\sup}) \mathbb{M} = \sup(C) \) and \( (C_{\beta+1}^{\text{Sel}}) C_{\beta+1} \in \Omega \mathbb{P}(\text{Sel}, C, C_{\beta}) \) for all \( \beta \), where \( \beta + 1 < \kappa \) and \( \kappa \) is the length of the transfinite sequence \( C \).

Thus, \((\mathbb{P}, \mathbb{M}, \text{Sel})\)-chains are monotonically increasing sequences \( C \) of interpretations of \( \mathbb{P} \), that reproduce themselves by iterating \( \Omega \mathbb{P} \). Note that this definition is non-constructive.

The main concept of the selector semantics is fixed in the following definition.

Definition 5. A model \( \mathbb{M} \) of a program \( \mathbb{P} \subseteq \mathbb{GLP} \) is \textit{Sel-generated} if and only if there exists a \((\mathbb{P}, \mathbb{M}, \text{Sel})\)-chain \( C \). The \textit{Sel-semantics} of the program \( \mathbb{P} \) is the set \( \text{Mod}_{\text{Sel}}(\mathbb{P}) \) of all \text{Sel-generated} models of \( \mathbb{P} \).

Example 2. The program \( \mathbb{P} = \{ \Rightarrow a, a \Rightarrow b, (a \lor \neg c) \land (c \lor \neg a) \Rightarrow c \} \) has the only \text{Sel}_l-generated model \( \{a, b, c\} \), namely via the chain \( C_1 = (\emptyset \xrightarrow{1} a \xrightarrow{2} \{a, c\}) \), where the rules applied in each step are denoted above the arrows. \( \{a, b\} \) and \( \{a, b, c\} \) are \text{Sel}_{lu}\)-generated (and \text{Sel}_c-generated) models, namely via the chains \( C_2 = (\emptyset \xrightarrow{1} a \xrightarrow{2} \{a, b\}) \) and \( C_1 \). \( \{a, b\} \) is the only \text{Sel}_c-generated model of \( \mathbb{P} \), namely via \( C_2 \).

Some properties of semantics generated by the selectors in Example 1 were studied in [15]. In Section 5, we will make use of the following results from [15].
Theorem 1 ([16]).
1. For definite programs \( P \subseteq \text{DLP} \), the unique element contained in \( \text{Mod}_i(P) = \text{Mod}_c(P) = \text{Mod}_I(P) \) is the least model of \( P \).
2. For normal programs \( P \subseteq \text{NLP} \), the unique element of \( \text{Mod}_i(P) \) is the inflationary model of \( P \) (as introduced in [12]).
3. For normal programs \( P \subseteq \text{NLP} \), the set \( \text{Mod}_i(P) = \text{Mod}_c(P) = \text{Mod}_I(P) \) contains exactly all stable models of \( P \) (as defined in [5]).
4. For disjunctive programs \( P \subseteq \text{DLP}^+ \), the minimal elements in \( \text{Mod}_i(P) = \text{Mod}_c(P) = \text{Mod}_I(P) \) are exactly all stable models of \( P \) (as defined in [13]), but for generalized programs \( P \subseteq \text{GLP} \), the sets \( \text{Mod}_i(P), \text{Mod}_c(P), \text{Mod}_I(P) \) may differ.
5. For generalized programs \( P \subseteq \text{GLP} \), \( \text{Mod}_i(P) \) is the set of stable generated models of \( P \) (as defined in [7]).

This shows that the framework of selector semantics covers some of the most important declarative semantics for normal logic programs. Selector generated models provide a natural extension of these semantics to generalized logic programs and allow systematic comparisons of many new and well-known semantics.

4 Selector generated models via level mappings

In [8, 10, 11], a uniform approach to different semantics for logic programs was given, using the notion of level mapping, as follows.

Definition 6. A level mapping for a logic program \( P \subseteq \text{GLP} \) is a function \( l : B_P \rightarrow \alpha \), where \( \alpha \) is an ordinal.

In order to display the style of level-mapping characterizations for semantics, we cite two examples which we will further discuss later on.

Theorem 2 ([11]). Every definite program \( P \subseteq \text{LP} \) has exactly one model \( M \), such that there exists a level mapping \( l : B_P \rightarrow \alpha \) satisfying

\[
\text{(Fd)} \quad \text{for every atom } a \in M \text{ there exists a rule } \bigwedge_{b \in B} b \Rightarrow a \in P \text{ such that } B \subseteq M \text{ and } \max \{ l(b) \mid b \in B \} < l(a).
\]

Furthermore, \( M \) coincides with the least model of \( P \). \( \square \)

Theorem 3 ([4]). Let \( P \) be a normal program and \( M \) be an interpretation for \( P \). Then \( M \) is a stable model of \( P \) iff there exists a level mapping \( l : B_P \rightarrow \alpha \) satisfying

\[
\text{(Fs)} \quad \text{for each atom } a \in M \text{ there exists a rule } \text{head}(r) = a, \text{body}(r)^+ \subseteq M, \text{body}(r)^- \cap M = \emptyset, \text{ and } \max \{ l(b) \mid b \in \text{body}(r)^+ \} < l(a). \quad \square
\]

It is evident, that among the level mappings satisfying the respective conditions in Theorems 2 and 3, there exist pointwise minimal ones.

We set out to prove a general theorem which characterizes selector generated models by means of level mappings, in the style of the results displayed above. The following notion will ease notation considerably.
Definition 7. For a level mapping \( l : B_P \rightarrow \alpha \) for a program \( P \subseteq \text{GLP} \) and an interpretation \( M \subseteq B_P \), the elements of the (transfinite) sequence \( C^{l,M} \) consisting of interpretations of \( P \) are for all \( \beta < \alpha \) defined by

\[
C_{\beta}^{l,M} = \{ a \in M \mid l(a) < \beta \} = M \cap \bigcup_{\gamma < \beta} l^{-1}(\gamma).
\]

Remark 1. Definition 7 implies that
1. the (transfinite) sequence \( C^{l,M} \) is monotonically increasing,
2. \( C_{0}^{l,M} = \emptyset \), and
3. \( M = \bigcup_{\beta < \alpha} C^{l,M}_{\beta} = \sup C^{l,M} \).

The following Theorem provides a mutual translation between the definition of selector semantics and a level mapping characterization.

Theorem 4. Let \( P \subseteq \text{HDLP} \) be a head disjunctive program and \( M \in \text{I}_P \). Then \( M \) is a \( \text{Sel} \)-generated model of \( P \) iff there exists a level mapping \( l : B_P \rightarrow \alpha \) satisfying the following properties.

\begin{enumerate}[\textbf{(L1)}]
  \item \( M = \sup (C^{l,M}) \in \text{Mod}(P) \).
  \item \( \text{For all } \beta \text{ with } \beta + 1 < \alpha \text{ we have } C^{l,M}_{\beta+1} \setminus C^{l,M}_{\beta} \in \text{Min}\left\{ J \in \text{I}_P \mid J \models \text{head} \left( R \left( C^{l,M}_{\beta}, J \right) \right)^+ \right\} \), where
    \[
    R \left( C^{l,M}_{\beta}, J \right) = \left\{ r \in \text{fire} \left( P, \text{Sel} \left( C^{l,M}, C^{l,M}_{\beta} \right) \right) \mid C^{l,M}_{\beta} \not\models \text{head} (r)^+ \text{ and } J \cup C^{l,M}_{\beta} \not\models \text{head} (r)^- \right\}.
    \]
  \item \( \text{For all limit ordinals } \lambda < \alpha \text{ we have } C^{l,M}_{\lambda} = \bigcup_{\beta < \lambda} C^{l,M}_{\beta} \).
\end{enumerate}

The proof of Theorem 4 is omitted for space limitations. It is rather involved and technical, and can be found in detail in [9].

Remark 2. As \( P \) is a head disjunctive program, we have \( C^{l,M}_{\beta} \not\models \text{head} (r)^+ \) if and only if \( \text{head} (r)^+ \cap C^{l,M}_{\beta} = \emptyset \), and \( J \cup C^{l,M}_{\beta} \not\models \text{head} (r)^- \) if and only if \( \text{head} (r)^- \subseteq J \cup C^{l,M}_{\beta} \), thus

\[
R \left( C^{l,M}_{\beta}, J \right) = \left\{ r \in \text{fire} \left( P, \text{Sel} \left( C^{l,M}, C^{l,M}_{\beta} \right) \right) \mid \text{head} (r)^+ \cap C^{l,M}_{\beta} = \emptyset \text{ and head} (r)^- \subseteq J \cup C^{l,M}_{\beta} \right\}.
\]

Also note that for every rule \( r \in \text{fire} \left( P, \text{Sel} \left( C^{l,M}, C^{l,M}_{\beta} \right) \right) \setminus R \left( C^{l,M}_{\beta}, J \right) \), we have \( J \cup C^{l,M}_{\beta} \not\subseteq \text{Mod} \left( \text{head} (r)^- \right) \) or \( C^{l,M}_{\beta} \not\subseteq \text{Mod} \left( \text{head} (r)^+ \right) \). Thus all of these rules are satisfied in the interval \( \left[ C^{l,M}_{\beta}, C^{l,M}_{\beta} \cup J \right] \).

For all selectors \( \text{Sel} \), it was shown in [15] that the Sel-semantics of programs in GLP is invariant with respect to the following transformations: the replacement \((\rightarrow_{eq})\) of the body and the head of a rule by logically equivalent formulae and the
splitting \( \rightarrow_{\text{hs}} \) of conjunctive heads, more precisely the replacement \( P \cup \{ \varphi \Rightarrow \psi \wedge \psi' \} \rightarrow_{\text{hs}} P \cup \{ \varphi \Rightarrow \psi, \varphi \Rightarrow \psi' \}. \)

Since every formula head \( \text{head}(r) \) is logically equivalent to a formula in conjunctive normal form, each selector semantics \( \text{Mod}_{\text{Sel}} \) of a generalized program \( P \) is equivalent to the selector semantics \( \text{Mod}_{\text{Sel}} \) of all head disjunctive programs \( Q \) where \( P \rightarrow^{*}_{\text{eq,hs}} Q \). Note that in the transformation \( \rightarrow^{*}_{\text{eq,hs}} \), no shifting of subformulas between the body and the head of a rule is involved. Therefore, Theorem 4 immediately generalizes to our main result.

**Corollary 1.** Let \( P \) be a generalized program and \( M \) an interpretation of \( P \). Then \( M \) is a \( \text{Sel} \)-generated model of \( P \) iff for any head disjunctive program \( Q \) with \( P \rightarrow^{*}_{\text{eq,hs}} Q \) there exists a level mapping \( l : \mathbb{B}_Q \rightarrow \alpha \) satisfying (L1), (L2) and (L3) of Theorem 4. \( \Box \)

5 Corollaries

We can now apply Theorem 4 in order to obtain level mapping characterizations for every semantics generated by a selector, in particular for those semantics generated by the selectors defined in Example 1 and listed in Theorem 1. For syntactically restricted programs, we can furthermore simplify the properties (L1), (L2) and (L3) in Theorem 4. Alternative level mapping characterizations for some of these semantics were already obtained directly in [11].

**Programs with positive disjunctions in all heads**

For rules \( r \in \text{HDLP} \), where \( \text{head}(r) \) is a disjunction of atoms, we have \( \text{head}(r)^- = \emptyset \). Hence we have \( \text{head}(r)^- \subseteq I \), i.e. \( I \models \text{head}(r)^- \), for all interpretations \( I \in \mathcal{I}_P \).

Thus the set \( R \left( C^i,M_{\beta}, J \right) \) from (L2) in Theorem 4 can be specified by

\[
R \left( C^i,M_{\beta}, J \right) = \left\{ r \in \text{fire} \left( P, \text{Sel} \left( C^{i,M}, C^{i,M}_{\beta} \right) \right) \mid C^i,M_{\beta} \not\models \text{head}(r)^+ \right\}.
\]

We furthermore observe that the set \( R \left( C^i,M_{\beta}, J \right) \) does not depend on the interpretation \( J \), so we obtain

\[
R' \left( C^i,M_{\beta} \right) = \left\{ r \in \text{fire} \left( P, \text{Sel} \left( C^{i,M}, C^{i,M}_{\beta} \right) \right) \mid C^i,M_{\beta} \not\models \text{head}(r)^+ = \emptyset \right\}
\]

and hence

\[
\text{Min} \left\{ J \in \mathcal{I}_P \mid J \models \text{head} \left( R \left( C^{i,M}_{\beta}, J \right) \right)^+ \right\} = \text{Min} \left( \text{Mod} \left( \text{head} \left( R' \left( C^{i,M}_{\beta} \right) \right) \right) \right).
\]

Thus for programs containing only rules whose heads are disjunctions of atoms we can rewrite condition (L2) in Theorem 4, as follows:

(L2d) for every \( \beta \) with \( \beta + 1 < \alpha \):

\[
C^{i,M}_{\beta+1} \setminus C^{i,M}_{\beta} \in \text{Min} \left( \text{Mod} \left( \text{head} \left( R' \left( C^{i,M}_{\beta} \right) \right) \right) \right), \text{where}
\]

\[
R' \left( C^{i,M}_{\beta} \right) = \left\{ r \in \text{fire} \left( P, \text{Sel} \left( C^{i,M}, C^{i,M}_{\beta} \right) \right) \mid C^{i,M}_{\beta} \not\models \text{head}(r)^+ = \emptyset \right\}.
\]
Programs with atomic heads

Single atoms are a specific kind of disjunctions of atoms. Hence for programs with atomic heads we can replace condition (L2) in Theorem 4 by (L2d), and further simplify it as follows.

For rules with atomic heads we have \(\text{head}\left(\{r \in P \mid \text{head}(r) \notin I\}\right) = \text{head}(P) \setminus I\) and therefore

\[
\text{head}\left(R'\left(C^M_{\beta}\right)\right)
\]

\[
= \text{head}\left(\{r \in \text{fire}\left(P, \text{Sel}\left(C^M_{\beta}, C^M_{\beta}\right)\right) \mid \text{head}(r) \cap C^M_{\beta} = \emptyset\}\right)
\]

\[
= \text{head}\left(\{r \in \text{fire}\left(P, \text{Sel}\left(C^M_{\beta}, C^M_{\beta}\right)\right) \mid \text{head}(r) \notin C^M_{\beta}\}\right)
\]

\[
= \text{head}\left(\text{fire}\left(P, \text{Sel}\left(C^M_{\beta}, C^M_{\beta}\right)\right)\right) \setminus C^M_{\beta}.
\]

Because all formulae in \(\text{head}(P)\) are atoms we obtain

\[
\text{Min}\left(\text{Mod}\left(\text{head}\left(R'\left(C^M_{\beta}\right)\right)\right)\right) = \text{Min}\left(\uparrow \left(\text{head}\left(R'\left(C^M_{\beta}\right)\right)\right)\right)
\]

\[
= \{\text{head}\left(R'\left(C^M_{\beta}\right)\right)\}
\]

and this allows us to simplify (L2) in Theorem 4 to the following:

(L2a) for each \(\beta\) with \(\beta + 1 < \alpha\):

\[
C^M_{\beta+1} \setminus C^M_{\beta} = \text{head}\left(\text{fire}\left(P, \text{Sel}\left(C^M_{\beta}, C^M_{\beta}\right)\right)\right) \setminus C^M_{\beta}.
\]

Inflationary models From Section 3 we know that for normal programs \(P\) the selector \(\text{Sel}_l\) generates exactly the inflationary model of \(P\) as defined in [12]. The generalizations of the definition of inflationary models and this result to head atomic programs are immediate. From [16] we also know that every \(\text{Sel}_l\)-generated model is generated by a \((P, M, \text{Sel}_l)\)-chain of length \(\omega\). Thus level mappings \(l : B_P \to \omega\) are sufficient to characterize inflationary models of head atomic programs. In this case, condition (L3) applies only to the limit ordinal \(0 < \omega\). But by remark 1, all level mappings satisfy this property. Therefore we do not need condition (L3) in the characterization of inflationary models.

Using Theorem 4 and the considerations above, we obtain the following characterization of inflationary models.

Corollary 2. Let \(P \subseteq \text{HALP}\) be a head atomic program and \(M\) be an interpretation for \(P\). Then \(M\) is the inflationary model of \(P\) iff there exists a level mapping \(l : B_P \to \omega\) with the following properties.

(L1) \(M = \sup (C^M) \in \text{Mod}(P)\).

(L2i) for all \(n < \omega\): \(C^M_{n+1} \setminus C^M_n = \text{head}\left(\text{fire}\left(P, C^M_n\right)\right) \setminus C^M_n\).

\(\square\)
Normal programs

For normal programs, the heads of all rules are single atoms. Hence the simplification \((L2a)\) of condition \((L2)\) in Theorem 4 applies for all selector generated semantics for normal programs.

The special structure of the bodies of all rules in normal programs allows an alternative formulation of \((L2a)\). In every normal rule, the body is a conjunction of literals. Thus for any set of interpretations \(J\) we have \(J \models \text{body}(r)\) iff \(\text{body}(r)^+ \subseteq J\) and \(\text{body}(r)^- \cap J = \emptyset\) for all interpretations \(J \in J\).

Stable models

We develop next a characterization for stable models of normal programs, as introduced in [5]. The selector \(\text{Sel}_{lu}\) generates exactly all stable models for normal programs. In [16], it was also shown that all \(\text{Sel}_{lu}\)-generated models \(M\) of a program \(P\) are generated by a \((P, M, \text{Sel})\)-chain of length \(\leq \omega\). So for the same reasons as discussed for inflationary models, level mappings with range \(\omega\) are sufficient to characterize stable models and condition \((L3)\) can be neglected.

For a normal rule \(r\) and two interpretations \(I, M \in \text{I}_P\) with \(I \subseteq M\) we have \(\{I, M\} \models \text{body}(r)\), i.e. \(I \models \text{body}(r)\) and \(M \models \text{body}(r)\), iff \(\text{body}(r)^+ \subseteq I\) and \(\text{body}(r)^- \cap M = \emptyset\). Combining this with \((L2a)\) we obtain the following characterization of stable models for normal programs.

**Corollary 3.** Let \(P \subseteq \text{NLP}\) be a normal program and \(M\) an interpretation for \(P\). Then \(M\) is a stable model of \(P\) iff there exists a level mapping \(l : \text{B}_P \rightarrow \omega\) satisfying the following properties:

\((L1)\) \(M = \sup (C_l^M) \in \text{Mod}(P)\).
\((L2s)\) for all \(n < \omega\):

\[C_{n+1}^M \setminus C_n^M = \text{head} \left( \{ r \in P \mid \text{body}(r)^+ \subseteq C_n^M, \text{body}(r)^- \cap M = \emptyset \} \right) \setminus C_n^M.\]

Comparing this with Theorem 3, we note that both theorems characterize the same set of models. Thus for a model \(M\) of \(P\) there exists a level mapping \(l : \text{B}_P \rightarrow \omega\) satisfying \((L1)\) and \((L2s)\) iff there exists a level mapping \(l : \text{B}_P \rightarrow \alpha\) satisfying \((Fs)\). The condition imposed on the level mapping in Theorem 3, however, is weaker than the condition in Corollary 3, because level mappings defined by \((P, M, \text{Sel})\)-chains are always pointwise minimal.

Definite programs

In order to characterize the least model of definite programs, we can further simplify condition \((L2)\) in Theorem 4. Definite programs are a particular kind of head atomic programs. For definite programs, the inflationary and the least model coincide. We can replace condition \((L2)\) in Theorem 4 by \((L2i)\) in Corollary 2. Since the body of every definite rule is a conjunction of atoms we obtain

\[\text{fire}(P, I) = \{ r \in P \mid \text{body}(r)^+ \subseteq I \}\]

for every interpretation \(I \in \text{I}_P\). Thus we get the following result.
Corollary 4. Let $P \subseteq \mathbb{LP}$ be a definite program and let $M$ be an interpretation for $P$. Then $M$ is the least model of $P$ iff there exists a level mapping $l : \mathcal{B}_P \rightarrow \omega$ satisfying the following conditions.

(L1) $M = \sup \left( C^{l,M}_n \right) \in \text{Mod}(P)$.

(L2l) For all $n < \omega$: $C^{l,M}_{n+1} \setminus C^{l,M}_n = \text{head} \left( \{ r \in P \mid \text{body}(r)^+ \subseteq C^{l,M}_n \} \right) \setminus C^{l,M}_n$ \quad \square

Comparing this to Theorem 2, we note that the relation between the conditions (L2l) and (Fd) are similar to those of the conditions (Fs) und (L2s).

6 Conclusions and Further Work

Our main result, Corollary 1 respectively Theorem 4 in Section 4, provides a characterization of selector generated models — in general form — by means of level mappings in accordance with the uniform approach proposed in [8, 10, 11]. As corollaries from this theorem, we have also achieved level mapping characterizations of several semantics encompassed by the selector generated approach due to [15, 16].

Our contribution is technical, and provides a first step towards a comprehensive comparative study of different semantics of logic programs under extended syntax by means of level mapping characterizations. Indeed, a very large number of syntactic extensions for logic programs are currently being investigated in the community, and even for some of the less fancy proposals there is often no agreement on the preferable way of assigning semantics to these constructs.

A particularly interesting case in point is provided by disjunctive and extended disjunctive programs, as studied in [6]. While there is more or less general agreement on an appropriate notion of stable model, as given by the notion of answer set in [6], there exist various different proposals for a corresponding well-founded semantics, see e.g. [17]. We expect that recasting them by means of level-mappings will provide a clearer picture on the specific ways of modelling knowledge underlying these semantics.

Eventually, we expect that the study of level mapping characterizations of different semantics will lead to methods for extracting other, e.g. procedural, semantic properties from the characterizations, like complexity or decidability results.

References


