Graph theoretical structures in logic programs and default theories

Yannis Dimopoulos\textsuperscript{a},*, Alberto Torres\textsuperscript{b}

\textsuperscript{a} Max-Planck-Institut für Informatik, Im Stadtwald, 66123 Saarbrücken, Germany
\textsuperscript{b} Stanford University, Computer Science Department, Stanford, CA 94305-2140, USA

Received August 1994; revised September 1995
Communicated by G. Levi

Abstract

In this paper we present a graph representation of logic programs and default theories. We show that many of the semantics proposed for logic programs with negation can be expressed in terms of notions emerging from graph theory, establishing in this way a link between the fields. Namely the stable models, the partial stable models, and the well-founded semantics correspond respectively to the kernels, semikernels and the initial acyclic part of an associated graph. This link allows us to consider both theoretical (existence, uniqueness) and computational problems (tractability, algorithms, approximations) from a more abstract and rather combinatorial point of view. It also provides a clear and intuitive understanding about how conflicts between rules are resolved within the different semantics. Furthermore, we extend the basic framework developed for logic programs to the case of Default Logic by introducing the notions of partial, deterministic and well-founded extensions for default theories. These semantics capture different ways of reasoning with a default theory.

1. Introduction

Humans often use patterns of reasoning that enable them to draw conclusions under incomplete information. These conclusions are retractable since new information can invalidate them. Much research in Nonmonotonic Reasoning has concentrated on capturing these patterns of reasoning in various formal representations. One of the most prominent nonmonotonic reasoning formalizations is Default Logic. On the other hand, recent developments in Logic Programming and deductive databases have shown that negation as failure is strongly related to various nonmonotonic formalisms, and in particular to default logic ([5]). Thus logic programs with negation provide us with a framework for nonmonotonic reasoning.

* Corresponding author. E-mail: yannis@mpi-sb.mpg.de.
Some recent work has dealt with the relation between some nonmonotonic formalisms and graph-theoretic constructs. Torres shows in [41] that the stable models of logic programs correspond to the kernels of an associated graph. This result is extended in [43], where it is proved that the maximal semikernels of the same graph correspond to partial stable models. For disjunction-free default theories, Dimopoulos and Magirou show in [13] that extensions correspond to kernels in a related graph. In this paper, we further extend the aforementioned results with the introduction of a unified semantic and graph-theoretic framework for logic programs and default theories.

We introduce the class of negative logic programs and a simple graph representation, the rule graph. We show that some of the most important proposals for defining the semantics of logic programs can be defined in terms of graph-theoretic structures in the rule graph. Stable models [21] correspond to kernels, partial stable models [33, 39] to semikernels and the well-founded partial model [45] to a special semikernel called the initial acyclic part. While for negative logic programs the translation to graphs is purely syntactic, in the case of general logic programs the translation uses, in a limited way, the semantics of the program. We use the logic programming notion of support, which we extend to disjunction-free default theories, to show that the above equivalences remain valid in the case of disjunction-free default theories. Finally, we extend the above mentioned semantics to the full case of the propositional default logic, by introducing the notions of the partial, deterministic and well-founded extensions of a default theory.

Aside from the theoretical interest of the above results, we believe the practical contribution of this paper is twofold. On the one hand, known properties of graph kernels and semikernels can improve our understanding of logic programming and default logic. Graphs give us an intuitive representation of the interactions between the rules and the different ways they can be resolved. Furthermore they allow us to approach the formalizations in a way that ignores the logical meaning and concentrates on their structural properties. It will become evident later in this paper that this is particularly useful when we try to investigate complexity issues or tackle problems like the existence of semantics (stable models, extensions) and the development of algorithms.

On the other hand, the unified graph model gives us a clear intuitive understanding about the translation of the semantical constructs of logic programs into the domain of default logic. The graph structures defined for logic programs remain meaningful in default logic. The proposed semantics for logic programs can be naturally transferred to default logic, and allow us to resolve various shortcomings of the initial semantics of default logic.

The rest of this paper is organized as follows. In Section 2, we introduce the fundamental concepts and results from logic programming, default logic and graph theory that we use in the rest of the paper. In Section 3, we introduce the restricted class of negative logic programs and prove the basic results of our graph model. In Section 4, we extend the results of the previous sections to the class of general logic programs. In Section 5, we explore some of the complexity and algorithmic implications of the graph model. In Section 6, we show how the semantical and graph-theoretical constructs can
be transferred to the case of default logic. Finally, in Section 7, we summarize the main contributions of this work.

2. Preliminaries

In this section we introduce the basic terminology and notation for logic programs, default theories and directed graphs used throughout this paper. We also summarize some of the fundamental results used in later sections.

The semantics presented in this section is along the lines of the hypothetical semantics for logic programs (see e.g. [23, 20, 27, 42]).

2.1. Logic programs and hypotheses

A program \( P \) is a set of first order rules of the form

\[
\alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_n \leftarrow \gamma_1 \wedge \gamma_2 \wedge \cdots \wedge \gamma_m
\]

where \( n \geq 1, m \geq 0 \), every \( \alpha_i \) is an atom, and every \( \gamma_i \) is a literal. The literals in the body of a rule are called subgoals. The above form of logic program differs from the standard since it allows conjunctions in the head of rules instead of single atoms. We choose to permit conjunctions to make a clearer connection with the default theories introduced in Section 6. Nevertheless, the above rule form should be seen solely as a shorthand for the set of rules

\[
\begin{align*}
\alpha_1 & \leftarrow \gamma_1 \wedge \gamma_2 \wedge \cdots \wedge \gamma_m \\
\alpha_2 & \leftarrow \gamma_1 \wedge \gamma_2 \wedge \cdots \wedge \gamma_m \\
& \vdots \\
\alpha_n & \leftarrow \gamma_1 \wedge \gamma_2 \wedge \cdots \wedge \gamma_m
\end{align*}
\]

A rule with no subgoals is considered identical to the conjunction in its head. All variables are implicitly universally quantified. A datalog program is a program with no occurrences of function symbols. In this paper we refer exclusively to datalog programs.

If \( r \) is a rule then \( \text{head}(r) \) denotes the set of atoms in the head of \( r \), and \( \text{body}(r) \) denotes the set of literals in its body. If \( R \) is a set of rules then \( \text{body}(R) = \bigcup_{r \in R} \text{body}(r) \) and \( \text{head}(R) = \bigcup_{r \in R} \text{head}(r) \).

Let \( P \) be a logic program. We denote by \( \mathcal{H}(P) \) the Herbrand base of \( P \) and by \( P_{\text{inst}} \) the Herbrand instantiation of \( P \), that is, the ground program obtained by replacing the variables in \( P \) by terms in its Herbrand universe in all possible ways. An assumption is a ground negative literal in \( \neg \mathcal{H}(P) \), and a hypothesis is a set of assumptions. If \( \Lambda \) is a set of literals then \( \neg \Lambda \) is the set of literals corresponding to the negation of the elements in \( \Lambda \).
A hypothesis $\Delta$ enables a rule $r$ if all negative subgoals in $r$ are contained in $\Delta$, that is, $(\text{body}(r) - \Delta) \subseteq \mathcal{H}(P)$. The set of rules in a program $P$ enabled by a hypothesis $\Delta$ is denoted by $\text{enabled}(\Delta, P)$.

2.2. Supports and attacks

We denote by $P_\Delta$ the ground program resulting from deleting all assumptions in a given hypothesis $\Delta$ from the body of rules in $P_{\text{inst}}$, and $P^+_\Delta$ the program resulting from deleting all rules with negative subgoals from $P_\Delta$. Since $P^+_\Delta$ is a ground Horn program for any $\Delta$, deduction can be limited to forward application of the rules without loss of expressive power.

**Definition 2.1.** A hypothesis $\Delta$ is a support for an atom $\alpha$ in a program $P$ (denoted by $\Delta \overset{P}{\rightarrow} \alpha$) if $P^+_\Delta \models \alpha$. If $\Theta \subseteq \mathcal{H}(P)$, we write $\Delta \overset{P}{\rightarrow} \Theta$ if for all $\alpha \in \Theta$ we have $\Delta \overset{P}{\rightarrow} \alpha$. We denote by $\Delta \overset{P}{\rightarrow}$ the set of atoms supported by a hypothesis $\Delta$. Furthermore, a support $\Delta$ is minimal for $\alpha$ (denoted by $\Delta \overset{\text{min}, P}{\rightarrow} \alpha$) if no subset of $\Delta$ supports $\alpha$.

**Example 2.2.** Consider the following program $P_1$:

\[
\begin{align*}
p &\leftarrow \neg q \land \neg t \\
q &\leftarrow \neg p \land \neg t \\
r &\leftarrow \neg p \\
r &\leftarrow \neg r \\
t &\leftarrow s
\end{align*}
\]

\{\neg q, \neg t, \neg p\} supports $p$, and there are only two minimal supports for $r$: $\{\neg p\}$ and $\{\neg r\}$. Moreover, $t$ has no support in $P_1$ even though there is a rule with $t$ in its head.

Intuitively, a hypothesis supports an atom if the latter can be proved by applying the rules “forward”, assuming true all the negative atoms in the former. Notice that support is then a monotonic operator. Notice also that a minimal support corresponds to the leaves of a proof tree and therefore is finite. All conclusions supported by an assumption $\Delta$ are entailed by $\Delta \cup P$. However, the reciprocal of this statement is not true. For instance, in $P_1$ above, the hypothesis $\{\neg r\}$ does not support $p$ while $P_1 \cup \{\neg r\} \models p$.

**Definition 2.3.** A hypothesis $\Delta$ attacks another hypothesis $\Delta'$ in a program $P$ (denoted by $\Delta \overset{P}{\hookrightarrow} \Delta'$) if $\Delta \overset{P}{\rightarrow} \beta$ for some $\neg \beta \in \Delta'$. A hypothesis $\Delta$ minimally attacks another hypothesis $\Delta'$ in a program $P$ (denoted by $\Delta \overset{\text{min}, P}{\hookrightarrow} \Delta'$) if it is a minimal hypothesis.

---

1 For simplicity we assume that the only atoms of the language of $P$ are those that correspond to literals that occur in $P$.

2 Notation: We omit the superscript $P$ from the above notation as well as others introduced later when it is clear from the context. A min superscript over a binary relation always indicates the minimality of the left operand (with respect to set inclusion).
such that $\Delta \overset{p}{\rightarrow} \beta$ for some $-\beta \in \Delta'$. A hypothesis $\Delta$ is self-consistent in a program $P$ if it does not attack itself.

In the example above, $\{\neg t, \neg p\} \overset{P_1}{\rightarrow} \{\neg p, \neg q\}$. The hypothesis $\{\neg p, \neg t\}$ is self-consistent but $\{\neg p, \neg q, \neg t\}$ is not since $\{\neg p, \neg q, \neg t\} \overset{P_1}{\rightarrow} \{q, p\}$.

The notion of an unfounded assumption defined next, was introduced in [45].

**Definition 2.4.** An assumption $-\beta$ is unfounded with respect to a hypothesis $\Delta$ if for every $\Delta'$ such that $\Delta' \rightarrow \beta$ we have $\Delta \rightarrow \Delta'$. We denote by $\mathcal{U}_P(\Delta)$ the set of all unfounded assumptions w.r.t. $\Delta$ in program $P$.

2.3. Logic program semantics

A (Herbrand) interpretation $I$ for a program $P$ is a subset of $\mathcal{H}(P) \cup \neg \mathcal{H}(P)$ such that $I \cap \neg I = \emptyset$. We denote by $I^+$ and $I^-$ respectively $I \cap \mathcal{H}(P)$ and $I \cap \neg \mathcal{H}(P)$. We also denote by $\overline{I}$ the set $\mathcal{H}(P) - (I^+ \cup \neg I^-)$. We say that $\alpha \in \mathcal{H}(P)$ is defined in $I$ if $\alpha \in I^+ \cup \neg I^-$ and undefined if $\alpha \in \overline{I}$. An interpretation $I$ is total if $\mathcal{H}(P) = I^+ \cup \neg I^-$, otherwise it is partial. An atom $\alpha$ is true in $I$ if $\alpha \in I$, false if $\neg \alpha \in I$. An interpretation $I$ is a partial model for a program $P$ if $P \cup I$ is consistent. A model is a total partial model. Finally, the set of rules enabled by an interpretation $I$ is given by the definition $\text{enabled}(I, P) = \text{enabled}(I^-, P)$. We now define the semantical constructs explored in this paper:

**Definition 2.5.** Let $P$ be a program and $\Delta$ be a self-consistent hypothesis. The supported interpretation of $\Delta$ is $I_\Delta = \Delta \cup \Delta^\rightarrow$. We say that an interpretation is supported if it is the supported interpretation of some self-consistent hypothesis $\Delta$. We say that an interpretation $I$ is well-founded if $I$ is supported and $I^- \subseteq \mathcal{U}(I^-)$. A well-founded interpretation $I$ is complete if $I^- = \mathcal{U}(I^-)$.

For example, in $P_1$ of Example 2.2 the interpretation $I = \{p, \neg q, \neg t\}$ is well-founded but not complete since $\{\neg q, \neg t\} = I^- \cap \mathcal{U}_P(I^-) = \{\neg q, \neg s, \neg t\}$. The set $\{p, \neg q, \neg s, \neg t\}$ is a complete well-founded interpretation.

**Definition 2.6.** Let $P$ be a program and let $I$ be a supported interpretation. We say that $I$ is a:

**Stable model:** ([21]) if $I$ is total.

**Partial stable model:** ([39]) if $I$ is a maximal well-founded interpretation.

**Deterministic (partial) model:** ([39]) if $I$ is a complete well-founded interpretation and it is contained in every partial stable model.

**Well-founded (partial) model:** ([45]) if $I$ is the minimal deterministic model.

---

3 Notice that for two minimal attacks $\Delta'$ and $\Delta''$ on a hypothesis $\Delta$, $\Delta' \subseteq \Delta''$ may hold, if $\Delta' \overset{\text{min}, P}{\rightarrow} \Delta$ on a literal $\neg \alpha$ and $\Delta'' \overset{\text{min}, P}{\rightarrow} \Delta$ on a literal $\neg \beta$. 
The well-founded model coincides with the unique minimal complete well-founded interpretation.

For instance, $P_1$ above has one stable model ($\{\neg p, q, r, \neg s, \neg t\}$), two partial stable models ($\{\neg p, q, r, \neg s, \neg t\}$ and $\{p, \neg q, \neg s, \neg t\}$), and only one deterministic model, its well-founded model ($\{\neg s, \neg t\}$).

We now show that partial stable models and stable models are complete well-founded interpretations and that stable models are nothing but total partial stable models. 4

**Proposition 2.7.** Let $I$ be a partial stable model of $P$. Then $I$ is a complete well-founded interpretation.

**Proof.** If $I$ were not a complete well-founded interpretation then $I_{W_P(I^-)}$ would be a well-founded interpretation strictly containing $I$. □

**Lemma 2.8.** Let $I$ be a stable model of $P$. Then $I$ is a well-founded interpretation.

**Proof.** If $\neg \beta \in I^-$ and $A \rightarrow \neg \beta$ then $A \notin I^-$, because $I^-$ is self-consistent. Since $\Delta \cup \neg I^+ \neq \emptyset$ and $I$ is supported, $I^- \rightarrow \Delta$. Therefore, $\neg \beta \in \mathcal{U}_P(I^-)$ and $I^- \subseteq \mathcal{U}_P(I^-)$. □

**Proposition 2.9.** Let $I$ be a stable model of $P$. Then $I$ is a complete well-founded interpretation.

**Proof.** Consider any $\neg \beta$ in $\mathcal{U}_P(I^-)$. $\beta \notin I^+$ since otherwise $I^- \rightarrow \beta$ and $I^-$ would have to attack itself because $\neg \beta \in \mathcal{U}_P(I^-)$. Since $I$ is total, $\neg \beta \in I^-$ and therefore $\mathcal{U}_P(I^-) \subseteq I^-$. Since $I$ is a well-founded interpretation, $I^- = \mathcal{U}_P(I^-)$. □

Since every stable model is total, the above proposition implies the following corollary.

**Corollary 2.10.** Let $I$ be a stable model of $P$. Then $P$ is a partial stable model.

Finally, notice that all of the above semantical constructs define the meaning of a program depending exclusively on the support relation that the program defines. Therefore, two programs that define the same support relation ought to be treated as identical. This notion is captured by the following definition.

**Definition 2.11.** Two logic programs, $P_1$ and $P_2$, are support-equivalent if for every hypothesis $\Delta$, we have $\Delta \models P_1 \vartriangleleft \alpha$ if and only if $\Delta \models P_2 \vartriangleleft \alpha$.

Notice that if two programs are support-equivalent then the stable models, partial stable models, deterministic models and well-founded models are the same for both programs.

---

4 The proof of this latter fact was originally given in [39].
2.4. Default theories

In [36] Default Logic was introduced in order to augment first-order logic with a set of default assertions. In this paper we restrict ourselves to the case of propositional Default Theories. A (propositional) default theory is a pair $\Delta = (D, W)$, where $W$ is a finite set of propositions and $D$ is a finite set of default rules of the form $d = a : Mb_1 \ldots Mb_n \models w$, where $a, b_1, \ldots, b_n, w$ are arbitrary propositions. Proposition $a$ is called prerequisite of the rule $d$ (denoted as $\text{Prer}(d)$), the set of propositions $b_1, \ldots, b_n$ justifications (denoted as $\text{Just}(d)$) and the proposition $w$ consequent (denoted as $\text{Cons}(d)$). The default rules are roughly rules of inference stating the fact that if $a$ is provable and $b_1, \ldots, b_n$ are consistent, then $w$ is also provable.

The key concept in Default Logic is that of an Extension of a default theory, which is intuitively what can consistently be believed given $(D, W)$.

Definition 2.12. A set of propositions $E$ is an extension of a propositional default theory $\Delta = (D, W)$ iff $E = \bigcup_{i=0}^{\infty} E_i$, where $E_0 = W$ and $E_{i+1} = \text{Th}(E_i) \cup \{a : Mb_1 \ldots Mb_n \models w \in D, a \in E_i \text{ and } \neg b_j \notin E \text{ for } 1 \leq j \leq n\}$.

A default theory for which both $W$ and the prerequisite, justifications and consequents of the rules are conjunctions (sets) of literals is called conjunctive default theory (or disjunction-free default theory). A theory which contains no prerequisites in its rules is called prerequisite-free default theory.

2.5. Graphs and kernels

A directed graph or graph\(^5\) is a pair $(\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is a set and $\mathcal{E}$ is a subset of $\mathcal{V} \times \mathcal{V}$. Elements in $\mathcal{V}$ are called nodes and members of $\mathcal{E}$ are called edges. If $e = (v_1, v_2)$ is an edge we say that $e$ goes from $v_1$ to $v_2$. If $\mathcal{G}$ is a graph then $\mathcal{V}(\mathcal{G})$ denotes the set of nodes in $\mathcal{G}$ and $\mathcal{E}(\mathcal{G})$ denotes the set of its edges. An edge $(v,v') \in \mathcal{E}(\mathcal{G})$ is called symmetric if $(v',v) \in \mathcal{E}(\mathcal{G})$. A chord of a cycle $\gamma = n_1, \ldots, n_p, n_1$ is an edge $(n_i, n_j)$ with $j \neq i \pm 1$ (mod $p$).

If $\mathcal{G}$ is a graph and $v \in \mathcal{V}(\mathcal{G})$ we define $\Gamma_+(v) = \{v' | (v,v') \in \mathcal{E}(\mathcal{G})\}$, and $\Gamma_-(v) = \{v' | (v',v) \in \mathcal{E}(\mathcal{G})\}$. These definitions are extended to sets of nodes through the following equations: $\Gamma_+(\mathcal{V}) = \bigcup_{v \in \mathcal{V}} \Gamma_+(v)$, and $\Gamma_-(\mathcal{V}) = \bigcup_{v \in \mathcal{V}} \Gamma_-(v)$. The subscript $\mathcal{G}$ will be dropped from the above notation whenever clear from the context.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph, and $\mathcal{V}'$ a subset of $\mathcal{V}$. We denote by $\mathcal{E}/\mathcal{V}'$ the set $\mathcal{E} \cap \mathcal{V}' \times \mathcal{V}'$. The subgraph of $\mathcal{G}$ induced by $\mathcal{V}'$, denoted by $\mathcal{G}/\mathcal{V}'$, is the graph $(\mathcal{V}', \mathcal{E}/\mathcal{V}')$.

Definition 2.13. Given a directed graph $(\mathcal{V}, \mathcal{E})$, and a subset $\mathcal{V}'$ of $\mathcal{V}$, we say that $\mathcal{V}'$ is:

- **Independent:** if there are no edges between elements of $\mathcal{V}'$, i.e., if $\mathcal{E}/\mathcal{V}' = \emptyset$.

\(^5\) In this paper the term graph refers exclusively to directed graphs.
Dominant (dominating): if for all \( v \in V - V' \) there is a \( v' \in V' \) such that \((v', v) \in \mathcal{E}\), i.e., if \( V - V' \subseteq \Gamma^+_\mathcal{G}(V') \).

Semidominant: if for all \( v \in V - V' \), such that \((v, v') \in \mathcal{E}\) with \( v' \in V' \) then there is a \( v'' \in V' \) such that \((v'', v) \in \mathcal{E}\), i.e., \( \Gamma^-_\mathcal{G}(V') \subseteq \Gamma^+_\mathcal{G}(V') \).

Definition 2.14. Let \( \mathcal{G} \) be a graph, and \( \mathcal{X} \) a subset of \( V(\mathcal{G}) \). We say that \( \mathcal{X} \) is a kernel if it is independent and dominant. We also say that \( \mathcal{X} \) is a semikernel if it is independent and semidominant.\(^6\)

The following proposition follows trivially from the above definition:

Proposition 2.15. Let \( \mathcal{G} \) be a graph. If \( \mathcal{X} \) is a kernel of \( \mathcal{G} \) then \( \mathcal{X} \) is a maximal semikernel of \( \mathcal{G} \).

Definition 2.16. A graph \( \mathcal{G} \) is kernel-perfect if for every \( V' \subseteq V(\mathcal{G}) \) the graph \( \mathcal{G}/V' \) has a kernel.

Many sufficient properties for a graph to be kernel-perfect have been found. The classical results are included in the following proposition:\(^7\)

Proposition 2.17. A graph \( \mathcal{G} \) is kernel-perfect if it satisfies any of the following properties:

1. \( \mathcal{G} \) is acyclic. In this case the kernel is unique (von Neumann).
2. \( \mathcal{G} \) contains no odd-length cycle (Richardson).
3. \( \mathcal{G} \) is symmetric.
4. \( \mathcal{G} \) is transitive. In this case all kernels have the same cardinality (König).

Now we introduce the notion of initial acyclic part of a graph and prove that it exists and is unique for every graph.

Definition 2.18. Let \( \mathcal{G} = (V, \mathcal{E}) \) be a graph. We define the initial acyclic segment of \( \mathcal{G} \) to be a set of independent nodes \( V' \subseteq V \) such that it can be well-ordered in such a way that for every \( v \in V' \) we have \( \Gamma^-(v) \subseteq \Gamma^+(\{v' \in V' : v' < v\}) \). The initial acyclic part of a graph is its maximal initial acyclic segment.

Lemma 2.19. If \( \mathcal{I} \) is a set of initial acyclic segments of \( \mathcal{G} \) then \( \mathcal{I} = \bigcup_{S \in \mathcal{I}} S \) is an initial acyclic segment of \( \mathcal{G} \).

Proof. For any \( S \in \mathcal{I} \), let \( <_S \) be a well-order of \( S \) that complies with Definition 2.18. Let \( <_{\mathcal{I}} \) be any well-order of \( \mathcal{I} \), and let \( S_v = \min_{<_{\mathcal{I}}} \{ S \in \mathcal{I} : s \in S \} \) for any \( s \in \mathcal{I} \). We define \( < \) in \( \mathcal{I} \) such that \( s < s' \) if and only if \( S_v <_{\mathcal{I}} S_v' \) or if \( S_v = S_v' \) and \( s <_{S_v} s' \). It is easy to see that \( < \) is a well-order that complies with Definition 2.18.

\(^6\) Often symmetric to our definitions are used for kernels and semikernels (see for example [3]).

\(^7\) For a more extensive review of the area see [4].
Proposition 2.20. Every graph has a unique initial acyclic part.

Proof. Notice that the empty set is an initial acyclic segment for every graph. It follows from the previous lemma that the union of all initial acyclic segments is the unique initial acyclic part. □

Finally we show that every acyclic segment is a semikernel.

Proposition 2.21. If $S$ is an initial acyclic segment of $G$ then $S$ is a semikernel of $G$.

Proof. It is an easy ordinal induction to prove that $S$ is an independent set. To prove semidominance, notice that if $v \in \Gamma^-(S')$ then there is a $v' \in S'$ such that $v \in \Gamma^+(v')$. □

3. Negative logic programs and rule graphs

In this section we introduce the restricted class of negative logic programs. We also introduce the rule graph whose vertices correspond to rules and whose edges capture the notion of attack. We show that kernels in this graph correspond to stable models while semikernels correspond to well-founded interpretations.

Definition 3.1. A negative logic program is a logic program containing only rules of the form

$$\alpha_1 \land \alpha_2 \land \cdots \land \alpha_n \leftarrow \neg \beta_1 \land \neg \beta_2 \land \cdots \land \neg \beta_m$$

where $n \geq 1$, $m \geq 0$, and every $\alpha_i$ and every $\beta_i$ is a ground atom.

The following properties follow from the form of negative logic programs.

Proposition 3.2. Let $P$ be a negative logic program. Then $A \leftarrow \alpha$ if and only if $A$ enables a rule in $P$ such that $\alpha \in \text{head}(r)$.

Proof. Since all rules in $P$ have only negative subgoals, for any $A$, $P_\alpha^+$ contains only one rule $r'$ with $\text{head}(r') = \text{head}(r)$ and $\text{body}(r') = \emptyset$ for every rule $r$ in $P$ such that $\text{body}(r) \subseteq A$. Therefore, $P_\alpha^+ \models \alpha$ if and only if there is a rule $r$ in $P$ such that $\text{body}(r) \subseteq A$ and $\alpha \in \text{head}(r)$. □

Corollary 3.3. Let $P$ be a negative logic program. If $\Delta \models_P \alpha$ then there is a rule $r$ in $P$ such that $\Delta = \text{body}(r)$ and $\alpha \in \text{head}(r)$.

Corollary 3.4. Let $P$ be a negative logic program. An assumption $\neg \beta$ is unfounded w.r.t. $\Delta$ if and only if for every rule $r$ in $P$ such that $\beta \in \text{head}(r)$, $\Delta \not\rightarrow \text{body}(r)$. 
Definition 3.5. Let \( P \) be a negative logic program. We say that \( P \) is reduced if, for every two rules \( r_1 \) and \( r_2 \) in \( P \):

1. If \( \text{body}(r_1) = \text{body}(r_2) \) then \( r_1 = r_2 \).
2. If \( \text{head}(r_1) \cap \text{head}(r_2) \neq \emptyset \) and \( \text{body}(r_1) \subseteq \text{body}(r_2) \) then \( r_1 = r_2 \).

The central property of reduced negative logic programs is that the bodies of rules are exactly the minimal supports of atoms in the program.

Proposition 3.6. Let \( P \) be a reduced negative logic program. \( \Delta \overset{\text{min}}{\rightarrow}^{P} \alpha \) if and only if there is a rule \( r \) in \( P \) such that \( \Delta = \text{body}(r) \) and \( \alpha \in \text{head}(r) \).

Proof. The “only if” part follows from Corollary 3.3. To prove the “if” part, consider a rule \( r \) in \( P \) such that \( \alpha \in \text{head}(r) \) and \( \Delta = \text{body}(r) \). Then \( \Delta \overset{P}{\rightarrow} \alpha \). Now, if \( \Delta' \overset{\text{min}}{\rightarrow}^{P} \alpha \) and \( \Delta' \subseteq \Delta \), there is a rule \( r' \) such that \( \text{body}(r') = \Delta' \subseteq \Delta = \text{body}(r) \). Condition 2 of Definition 3.5 implies that \( \Delta = \Delta' \). □

Reduced negative logic programs can be seen as support-equivalent canonical forms for negative logic programs.\(^8\) The next theorem shows that a reduced negative logic program can in fact be obtained by “reducing” a given negative logic program.

Theorem 3.7. Let \( P \) be a negative logic program. There is a reduced program \( P' \) such that \( P' \) is support equivalent to \( P \). Moreover, given \( P \), \( P' \) can be computed in polynomial time.

Proof. Given a negative logic program \( P \) we can build a reduced negative logic program by using the following procedure:

(1) for each \( r \) in \( P \)  
    if \( \exists r' \in P (\text{body}(r') = \text{body}(r)) \) then  
      remove \( r \) from \( P \)  
      add \( \text{head}(r) \) to \( \text{head}(r') \)  
    end if  
  end for

(2) for each \( r \) in \( P \)  
    for each \( \alpha \) in \( \text{head}(r) \)  
      if \( \exists r' \in P (\alpha \in \text{head}(r') \land \text{body}(r') \subseteq \text{body}(r)) \) then  
        if \( \text{head}(r) - \{ \alpha \} \neq \emptyset \) then  
          remove \( \alpha \) from \( \text{head}(r) \)  
        else  
          remove \( r \) from \( P \)  
        end if  
      end if  
    end for

\(^8\) We generalize this result to general logic programs in Section 4.
Loop (1) collapses rules with the same body, and therefore $P$ satisfies Condition 1 of Definition 3.5 after the loop exit. Loop (2) removes redundant atoms and rules, and therefore the resulting $P$ satisfies Condition 2 of Definition 3.5. The above program can obviously be implemented in polynomial time. □

We now introduce a graph theoretical representation for reduced negative logic programs.

**Definition 3.8.** Let $P$ be a reduced negative logic program. The *rule graph* of $P$ (denoted by $\mathcal{R}(P)$) is the directed graph $(\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{r | r \in P\}$ and $\mathcal{E} = \{(r_1, r_2) | \text{head}(r_1) \cap \neg \text{body}(r_2) \neq \emptyset\}$.\(^9\)

In the following we introduce the main results of this section, linking the semantics introduced in the previous section to the graph theoretical structures of kernels and semikernels in the rule graph.

**Theorem 3.9.** Let $P$ be a reduced negative logic program. If $I$ is a well-founded interpretation of $P$ then $\text{enabled}(I, P)$ is a semikernel of $\mathcal{R}(P)$.

**Proof.** Since $I^-$ is self-consistent, $\text{enabled}(I, P)$ is an independent set. Now, if there is an edge $(r, r')$ in $\mathcal{R}(P)$ with $r' \in \text{enabled}(I, P)$ then $\text{body}(r)$ is a minimal support of an atom $\alpha$ such that $\neg \alpha \in \text{body}(r')$. Since $\neg \alpha \in I^-$, $\neg \alpha$ is unfounded w.r.t. $I^-$ and $I^- \not\Rightarrow \text{body}(r)$. Therefore, there is a rule $r''$ in $\text{enabled}(I, P)$ such that $\text{body}(r'') \not\Rightarrow \text{body}(r)$ and then $(r'', r)$ is an edge in $\mathcal{R}(P)$. □

**Theorem 3.10.** Let $P$ be a reduced negative logic program. If $K$ is a semikernel of $\mathcal{R}(P)$ then $\text{body}(K)$ is a well-founded interpretation of $P$.

**Proof.** First we prove that $\text{body}(K)$ is self-consistent. If $\text{body}(K)$ is not self-consistent then there is a rule $r$ in $\text{enabled}(\text{body}(K), P)$ such that $\alpha \in \text{head}(r)$ and $\neg \alpha \in \text{body}(K)$. Therefore, there is an edge from $r$ to some rule in $K$. Since $K$ is a semikernel then there is a rule $r' \in K$ such that there is an edge $(r', r)$ in $\mathcal{R}(P)$. But this means that $\text{body}(r')$ is a minimal attack of an assumption $\neg \beta$ in $\text{body}(r) \subseteq \text{body}(K)$. Since $\neg \beta \in \text{body}(K)$, there is an $r'' \in K$ such that $\neg \beta \in \text{body}(r'')$. Thus, there is an edge from $r'$ to $r''$, but this edge would contradict the supposition that $K$ is independent.

Now we have to prove that $\text{body}(K) \subseteq \Upsilon_P(\text{body}(K))$. Let $\neg \beta \in \text{body}(K)$ and $\Delta \not\Rightarrow^P \beta$. Then there is a rule $r$ in $P$ such that $\text{body}(r) = \Delta$. Therefore, there is an edge from $r$ to some rule in $K$, but since $K$ is a semikernel there is an edge from some other rule $r'$ in $K$ to $r$. Then $\text{body}(r') \not\Rightarrow^P \Delta$ and $\text{body}(K) \not\Rightarrow^P \Delta$. Therefore $\neg \beta$ is unfounded w.r.t. $\text{body}(K)$. □

\(^9\) Notice that $\mathcal{E} = \{(r_1, r_2) | \text{body}(r_1) \not\Rightarrow^P \text{body}(r_2)\}$. 
Theorem 3.10 is not the full reciprocal of Theorem 3.9 since there are well-founded interpretations that are not of the form $I_{\text{body}(K)}$, where $K$ is a semikernel of $\mathcal{R}(P)$. A well-founded interpretation can contain other assumptions that are either not explicitly used in the program or are heads of rules invalidated by the assumptions in the bodies of the rules of a semikernel. We now combine theorems 3.9 and 3.10 through the introduction of addition sets.

**Definition 3.11.** Let $P$ be a program and $A$ a hypothesis. A subset $T$ of $\mathcal{U}_P(A) - A$ is an addition set for $A$ in $P$ if $(A \cup T)^{\mathcal{P}} = A^{\mathcal{P}}$.

**Lemma 3.12.** Let $I$ be any interpretation of $P$. Then the following statements are true:

1. $(I^-)^{\mathcal{P}} = \text{body}(\text{enabled}(I,P))^{\mathcal{P}}$.
2. $\mathcal{U}_P(I^-) = \mathcal{U}_P(\text{body}(\text{enabled}(I,P)))$.

**Proof.** Proposition 1 is trivial, since the only assumptions in $I^-$ that can be used to apply rules are in $\text{enabled}(I,P)$. Proposition 2 follows from Proposition 1. □

**Theorem 3.13.** An interpretation $I$ for a reduced negative program $P$ is well-founded if and only if there is a semikernel $K$ in $\mathcal{R}(P)$ such that $I = I_{\text{body}(K)} \cup T$ where $T$ is an addition set for $\text{body}(K)$ in $P$.

**Proof.** We first prove the “if” part. Let $K$ be semikernel of $\mathcal{R}(P)$ and $T$ an addition set for $\text{body}(K)$ in $P$. By Theorem 3.10, we have that $I_{\text{body}(K)}$ is a well-founded interpretation. Now $I = I_{\text{body}(K)} \cup T$ is supported since the assumptions in $T$ do not support any new atom. And since $T \subseteq \mathcal{U}_P(\text{body}(K)) = \mathcal{U}_P(\text{body}(K) \cup T)$, $I$ is well-founded.

To prove the “only if” part, consider any well-founded interpretation $I$. Since $I$ is supported, $I = I^- \cup (I^-)^{\mathcal{P}} = \text{body}(\text{enabled}(I,P)) \cup T \cup (I^-)^{\mathcal{P}}$. By Lemma 3.12, $I = \text{body}(\text{enabled}(I,P)) \cup T \cup \text{body}(\text{enabled}(I,P))^{\mathcal{P}}$. By Theorem 3.9 we know that $\text{enabled}(I,P)$ is a semikernel of $\mathcal{R}(P)$. To prove that $T = I^- - \text{body}(\text{enabled}(I,P))$ is an addition set for $\text{body}(\text{enabled}(I,P))$ in $P$ we notice that since $I$ is well-founded we have that $T \subseteq \mathcal{U}_P(I^-) - \text{body}(\text{enabled}(I,P))$. By Lemma 3.12 we have $\mathcal{U}_P(I^-) = \mathcal{U}_P(\text{body}(\text{enabled}(I,P)))$, so $T \subseteq \mathcal{U}_P(\text{body}(\text{enabled}(I,P))) - \text{body}(\text{enabled}(I,P))$. By Lemma 3.12 we also have that $(I^-)^{\mathcal{P}} = \text{body}(\text{enabled}(I,P))^{\mathcal{P}}$. Therefore $T$ is an addition set for $\text{body}(\text{enabled}(I,P))$. □

**Corollary 3.14.** Let $P$ be a reduced negative logic program. An interpretation $I$ for $P$ is a partial stable model of $P$ if and only if there is a maximal semikernel $K$ in $\mathcal{R}(P)$ such that $I = I_{\text{body}(K)} \cup T$ where $T = \mathcal{U}_P(\text{body}(K)) - \text{body}(K)$ is the maximal addition set for $\text{body}(K)$ in $P$.

\footnote{In \cite{41,43} a different approach is used. We discuss this approach in Section 4.}
Stable models impose stronger restrictions on the rules they enable. While rules enabled by a well-founded interpretation need to counterattack only rules that attack them, the rules enabled by the stable models need to attack every other rule. The next theorem formally states the link between stable models and kernels.

**Theorem 3.15.** An interpretation $I$ for a reduced negative program $P$ is a stable model of $P$ if and only if there is a kernel $K$ in $\mathbb{R}(P)$ such that $I = I_{\text{body}(K)} \cup T$ where $T = \mathcal{U}_P(\text{body}(K)) - \text{body}(K)$ is the maximal addition set for $\text{body}(K)$ in $P$.

**Proof.** To prove the “if” part, notice that since $K$ is a kernel then it is also a maximal semikernel, so $I$ is a partial stable model. Furthermore, since $K$ is a kernel, then every atom is either in the head of an enabled rule (hence it is supported by $\text{body}(K)$) or all its rules are made invalid by $\text{body}(K)$ (hence it is unfounded w.r.t. $\text{body}(K)$). Therefore $I$ is total.

To prove the “only if” part, let $Z = I^- \cup (I^-)^+$ be a stable model and define $K = \text{enab}(I, P)$. Then $I$ is a partial stable model and by the previous corollary we know that $K$ is a maximal semikernel. We must prove that $K$ is a kernel. Since $K$ is independent, if $K$ is not a kernel then there is a rule $r \in P - K$ such that $r \notin \Gamma_{\mathbb{R}(P)}^+(K)$. The fact $r \notin \Gamma_{\mathbb{R}(P)}^+(K)$ means that for all the literals $-b \in \text{body}(r)$, $b \notin \text{enab}(I, P)^-$, because otherwise $r \in \text{enab}(I, P)$, hence $r \in K$. Therefore $m$ is undefined, a contradiction since $Z$ is total.

The next theorem demonstrates the fact that within well-founded models stronger restrictions are imposed on the way the interactions between the rules are resolved. In particular, well-founded models enable only rules that satisfy a non-circularity condition in the way they interact.

**Theorem 3.16.** Let $P$ be a reduced negative logic program. The interpretation $I$ is the well-founded model of $P$ if and only if $I = I_{\text{body}(IP)} \cup T$ where $IP$ is the initial acyclic part of $\mathbb{R}(P)$ and $T = \mathcal{U}_P(\text{body}(IP)) - \text{body}(IP)$ is the maximal addition set for $\text{body}(IP)$ in $P$.

**Proof.** To prove that if $I = I_{\text{body}(IP)} \cup T$ then $I$ is the minimal complete well-founded interpretation of $P$, we first show that $T$ is an addition set for $I_{\text{body}(IP)}$. Since $IP$ is maximal then $\forall r \in P$, if $\text{body}(r) \subseteq T \cup \text{body}(IP) = \mathcal{U}_P(\text{body}(IP)) \cup \text{body}(IP)$, then $r \in IP$ and $\text{body}(r) \subseteq \text{body}(IP)$. Hence, $(\mathcal{U}_P(\text{body}(IP)) \cup \text{body}(IP))^+ = \text{body}(IP)^+$ and $T$ is an addition set for $I_{\text{body}(IP)}$. Since the initial acyclic part is a semikernel, it follows from theorem 3.13 that $I$ is a well-founded interpretation. Additionally, since $T \subseteq I$, $I$ is complete. It remains to show that $I$ is the minimal such set. First observe that every literal of $T$ must be included in $I$ for $I$ to be complete. Additionally we can prove inductively on the well-order for $IP$ that if some literal $p \in \text{body}(IP)$ is omitted
Let I now be the the well-founded model of P. Let I_1 = \{-b | \exists r \in P, b \in \text{head}(r)\}. Since I is a complete well-founded interpretation then I_1 \subseteq I. Let R_1 = \{r | r \in P, \text{body}(r) \subseteq I_1\} = \text{enabled}(I_1, P). Then \text{head}(R_1) \subseteq I, and R_1 is an initial acyclic segment for \mathcal{R}_P(P). Let I_2 = \{-b | \forall r, b \in \text{head}(r), \exists p \in \text{body}(r), p \in \text{head}(R_1)\}. Then I_2 \subseteq \mathcal{U}_P(I) and hence I_2 \subseteq I. Let R_2 = \{r | r \in P, \text{body}(r) \subseteq I_1 \cup I_2\} = \text{enabled}(I_1 \cup I_2, P). Then again \text{head}(R_2) \subseteq I and R_1 \cup R_2 is an initial acyclic segment for \mathcal{R}_P(P). Iterating this way over the ordinals we define I' = I \cup \bigcup_{\alpha < \omega} R_{\alpha} = \text{enabled}(\bigcup_{I_n}) = \text{enabled}(\bigcup_{I_\alpha}) = \text{enabled}(\bigcup_{I_\alpha} R_{\alpha}). By lemma 3.12, \mathcal{U}_P(\bigcup_{I_n}) = \mathcal{U}_P(\text{body}(\text{enabled}(\bigcup_{I_n}))), which means \mathcal{U}_P(\bigcup_{I_n} R_{\alpha}) = \mathcal{U}_P(\text{body}(\bigcup_{I_n} R_{\alpha})). Hence I' = I_{\text{body}(\bigcup_{I_n} R_{\alpha})} \cup (\mathcal{U}_P(\text{body}(\bigcup_{I_n} R_{\alpha})) - \text{body}(\bigcup_{I_n} R_{\alpha})).

Note that I' \subseteq I and that \bigcup_{I_n} R_{\alpha} is an initial acyclic segment for \mathcal{R}_P(P). Furthermore, \bigcup_{I_n} R_{\alpha} is the maximal initial acyclic segment since for any rule r \in P - (\bigcup_{I_n} R_{\alpha}) there is a literal \neg b \in \text{body}(r) such that b \in \text{head}(r'), with r' \in P - (\bigcup_{I_n} R_{\alpha}). Hence, \bigcup_{I_n} R_{\alpha} is the initial acyclic part of \mathcal{R}_P(P). Then \bigcup_{I_n} R_{\alpha} is a semikernel, hence I' is a well-founded interpretation provided that T = \mathcal{U}_P(\text{body}(\bigcup_{I_n} R_{\alpha})) - \text{body}(\bigcup_{I_n} R_{\alpha}) is an addition set for \text{body}(\bigcup_{I_n} R_{\alpha}). Note that T \cup \text{body}(\bigcup_{I_n} R_{\alpha}) = \mathcal{U}_P(\text{body}(\bigcup_{I_n} R_{\alpha})) \cup \text{body}(\bigcup_{I_n} R_{\alpha}). Assume that a \in (\mathcal{U}_P(\text{body}(\bigcup_{I_n} R_{\alpha})) \cup \text{body}(\bigcup_{I_n} R_{\alpha}))^+ and there is a rule r \in P such that a \in \text{head}(r) and for every \neg b \in \text{body}(r), \neg b \in \mathcal{U}_P(\text{body}(\bigcup_{I_n} R_{\alpha})) \cup \text{body}(\bigcup_{I_n} R_{\alpha}). Then we can prove that r \in \bigcup_{I_n} R_{\alpha}, hence (T \cup \text{body}(\bigcup_{I_n} R_{\alpha}))^+ = (\text{body}(\bigcup_{I_n} R_{\alpha}))^+, that is, Y is an addition set. Hence I' is a well-founded interpretation. Furthermore, notice that I' is a total well-founded interpretation. Since I' \subseteq I and I is the minimal total well-founded interpretation, I = I'. Since \bigcup_{I_n} R_{\alpha} is the initial acyclic part of \mathcal{R}_P(P), I = I_{\text{body}(IP)} \cup (\mathcal{U}_P(\text{body}(IP)) - \text{body}(IP)).

We recapitulate the results presented in this section by means of the following example.

Example 3.17. Let P_2 be the following negative logic program:

\[
\begin{align*}
p, s &\leftarrow \neg q, \neg r & (r_1) \\
q &\leftarrow \neg p, \neg r & (r_2) \\
s &\leftarrow \neg s & (r_3) \\
t &\leftarrow \neg r & (r_4) \\
u &\leftarrow \neg t & (r_5)
\end{align*}
\]

The rule graph of P_2, \mathcal{R}_P(P_2), is depicted in Fig. 1. The semikernels of this graph are S_1 = \{r_1\}, S_2 = \{r_2\}, S_3 = \{r_4\}, S_4 = \{r_1, r_4\}, S_5 = \{r_2, r_4\} (the last two are maximal). The first three semikernels correspond to the well-founded interpretations I_1 = \{-q, \neg r, p, s, t\}, I_2 = \{-p, \neg r, q, t\}, I_3 = \{-r, t\}, respectively. None of these well-founded interpretations is complete. The corresponding complete well-founded models are I_4 = \{-q, \neg r, p, s, t, \neg u\}, I_5 = \{-p, \neg r, q, t, \neg u\}, I_6 = \{-r, t, \neg u\}. The first two of
these complete well-founded models correspond to the maximal semikernels $S_4$ and $S_5$ respectively and therefore are partial stable models. Since $S_4$ is a kernel for $\mathcal{KB}(P)$ then $I_4$ is also a stable model. Finally the initial acyclic part of $\mathcal{KB}(P)$ is the set $S_3$, hence the well-founded model of $P$ is the set $I_5 = \{\neg r, t, \neg u\}$.

4. The case of general logic programs

In this section, we extend the results for negative logic programs we have developed so far, to the case of general logic programs. We show that for every general logic program there is a support-equivalent reduced negative logic program. We also show that the rule graph of the corresponding negative program represents the support relation of the original program.

**Definition 4.1.** The negative equivalent of a given logic program $P$ is the negative logic program $P^-$ containing exactly every rule $r$ where $\text{body}(r)$ is a minimal support of some atom in $P$, and $\text{head}(r) = \{z|\text{body}(r) \xrightarrow{\text{min}_P} z\}$.

**Proposition 4.2.** Let $P$ be a logic program. Then $P^-$ is reduced and support-equivalent to $P$.

**Proof.** Since no two different rules having the same body in the transformed program fulfills Condition 1 of Definition 3.5. Now, if $z \in \text{head}(r_1) \cap \text{head}(r_2)$, it is not possible that $\text{body}(r_1) \subseteq \text{body}(r_2)$ because otherwise $\text{body}(r_2)$ would not be a minimal support of $z$. Therefore $P^-$ is reduced.

The fact that $P^-$ is support-equivalent to $P$ follows directly from Proposition 3.6. □
Proposition 4.3. If \( P \) is a datalog program, then \( P^- \) is finite.

Proof. Since \( P \) is a datalog program, then \( \mathcal{H}(P) \) is finite. But \( P^- \) can not contain more than \( 2^{\mathcal{H}(P)} \) rules. \( \square \)

We now introduce the minimal attack graph of a logic program and show that it corresponds to the rule graph of its negative equivalent.

Definition 4.4. The **minimal attack graph** of a program \( P \), denoted by \( \mathcal{MA}(P) \), is the directed graph \( (\mathcal{V}, \mathcal{E}) \), where \( \mathcal{V} = \{ \Delta | \exists x, a \overset{\text{min}}{\rightarrow} P(x) \} \) and \( \mathcal{E} = \{ (\Delta_1, \Delta_2) | \Delta_1 \overset{\text{min}}{\rightarrow} \Delta_2 \} \).

Proposition 4.5. Let \( P \) be a logic program. The graph \( \mathcal{MA}(P) \) is isomorphic to the graph \( \mathcal{HA}(P^-) \).

Proof. Consider the function that maps every minimal support in \( P \) into the rule with the same body in \( P^- \). It follows trivially from the above definition that this function is an isomorphism. \( \square \)

Combining the above proposition with the results in Section 3 we have the following corollary.

Corollary 4.6. Let \( P \) be a program, and let \( I \) be an interpretation for \( P \). The following propositions are true:
1. \( I \) is a well-founded interpretation of \( P \) if and only if there is a semikernel \( K \) in \( \mathcal{MA}(P) \) such that \( I = I_{\text{body}(K)} \cup T \) where \( T \) is an addition set for \( \text{body}(K) \) in \( P \).
2. \( I \) is a partial stable model of \( P \) if and only if there is a maximal semikernel \( K \) in \( \mathcal{MA}(P) \) such that \( I = I_{\text{body}(K)} \cup T \) where \( T = \mathcal{U}_P(\text{body}(K)) - \text{body}(K) \) is a maximal addition set for \( \text{body}(K) \) in \( P \).
3. \( I \) is a stable model of \( P \) if and only if there is a kernel \( K \) in \( \mathcal{MA}(P) \) such that \( I = I_{\text{body}(K)} \cup T \) where \( T = \mathcal{U}_P(\text{body}(K)) - \text{body}(K) \) is a maximal addition set for \( \text{body}(K) \) in \( P \).
4. \( I \) is the well-founded model of \( P \) if and only if \( I = I_{\text{body}(IP)} \cup T \) where \( IP \) is the initial acyclic part of \( \mathcal{MA}(P) \) and \( T = \mathcal{U}_P(\text{body}(IP)) - \text{body}(IP) \) is a maximal addition set for \( \text{body}(IP) \) in \( P \).

It is well known that stable models do not exist for every program. Using the results of Proposition 2.17 we can identify classes of programs for which the existence of a stable model is guaranteed by some property of its minimal attack graph.

Definition 4.7. A program \( P \) is **support-stratified** if \( \mathcal{MA}(P) \) is acyclic.

A consequence of the classical result by von Neumann is the following.
Proposition 4.8. Every support-stratified program has a unique stable model.

Theorem 4.9. Every locally stratified program is support-stratified.

Proof. Notice that if \( \neg \beta \) belongs to a minimal support of \( \alpha \) in \( P \), there is a proof of \( \alpha \) that uses a rule with \( \neg \beta \) in its body. It follows that \( \alpha \) depends on \( \neg \beta \). Therefore, if the dependency relation has no cycles the minimal support relation can not have cycles.

Example 4.10. Support-stratification is in fact a strict extension of local stratification. This fact is shown in the following example:

\[
p \leftarrow q \land r
q \leftarrow \neg p
\]

Notice that even though \( p \) depends on its negation, this dependency will never be used to prove \( p \) because \( r \) cannot be proved (there is no support for \( r \)).

By Proposition 2.17 we also obtain the following results.

Proposition 4.11. Let \( P \) be a logic program. If \( \mathcal{M}(P) \) is symmetric then \( P \) has at least one stable model.

Definition 4.12. A program \( P \) is odd-cycle free if every cycle in \( \mathcal{M}(P) \) is of even length.

Proposition 4.13. Every odd-cycle free program has at least one stable model.

The class of odd-cycle free program, even though slightly more general, roughly corresponds to the class of call-consistent programs which are known to have at least one stable model [28,37]. More general classes of kernel-perfect graphs have been found in the last years (see [4]). By applying the results in [19, 18] (for further results refer to [15, 24]) we obtain the next two propositions.

Proposition 4.14. If every odd cycle of the graph \( \mathcal{M}(P) \) of a logic program \( P \), has at least two symmetric edges, then \( P \) has at least one stable model.

Proposition 4.15. If every odd cycle of the graph \( \mathcal{M}(P) \) of a logic program \( P \), has two chords whose heads are consecutive nodes of the cycle, then \( P \) has at least one stable model.

It is important to note that both propositions concern kernel-perfect graphs, that is, not only \( \mathcal{M}(P) \) itself, but also every subgraph of \( \mathcal{M}(P) \) has at least one stable
model. The next theorem refers to the case of random graphs. The notation $D(n, p)$ denotes a directed graph on $n$ nodes with edge probability $p$.

**Theorem 4.16** (Fernadez de la Vega [16]). Let $p$ be fixed, $0 \leq p \leq 1$. The probability that the random directed graph $D(n, p)$ possesses a kernel tends to 1 as $n \to \infty$.

Finally, notice that the translation of a general logic program $P$ into a graph is not purely syntactic. This is because $P$ is translated into a negative logic program $P^-$ first, and this translation uses the notion of minimal support which is a semantic one. Nevertheless, the use of the semantics is limited since the notion of minimal support is common to all the semantics studied in this paper. Hence, while some semantic notion is used in the translation, this notion is independent of any particular semantics.

On the other hand, when transforming a logic program $P$ to a negative one $P^-$, we may end up with a program $P^-$ that is exponentially larger than the original $P$.

In [11], by applying the results of [13], another method for translating a normal logic program into a directed graph is presented. The translation is purely syntactic and the size of the graph is polynomial in the size of the logic program. We briefly introduce this method in the following.

Let $P$ be a general logic program. We define the **complete rule graph** of $P$ to be the graph $G_P = (N,E)$ where the set of nodes is $N = R \cup L$, with $R = \{r_i \mid r_i \text{ a rule of } P\}$ and $L = \{a_i \mid a_i \text{ occurs in } P\}$, whilst the set of edges is $E = \{(r_i, r_j) \mid \neg p \in \text{body}(r_j) \text{ and } p \in \text{head}(r_i)\} \cup \{(a_i, r_j) \mid a_i \in \text{body}(r_j)\} \cup \{(r_i, a_j) \mid a_j = \text{head}(r_i)\}$ (see [11], [12] for details).

We can prove that every stable model of $P$ corresponds to a kernel of $G_P$. Namely, if $M$ is a stable model for a program $P$, then there is a kernel $K$ for the rule graph $G_P$ such that for every $p \in M^+$ ($M^+$ denotes the set of positive literals in a set of literals $M$) there is a node $r_i$ in $K$ such that $\text{head}(r_i) = p$. However the converse is not true. This is because of possible circular support between the rules. It turns out that the models of the program's completion are in direct correspondence with the kernels of the program's complete rule graph.

5. **Complexity and algorithms**

The intractability of most of the nonmonotonic formalisms, even in very simple cases, is one of the central problems research in the field needs to tackle. In this section we show how graph theory can contribute in obtaining new complexity results and algorithms, determining cases where reasoning is tractable, and defining new notions of approximation.

To start with, recall that the problem of determining whether a negative logic program possesses stable models is NP-complete (see e.g. [13]). On the other hand

---

11 Related results can be found in [40].

12 The complexity results in [29] regarding autoepistemic logic also imply this result.
semikernels, or equivalently well-founded interpretations and partial stable models, always exist. For example every graph has a trivial semikernel which is the empty set. Hence one may expect better computational behavior in the case of semikernels. Furthermore since the existence of semikernels is guaranteed we need to formulate a slightly different decision problem.

**Decision problem:** Instance: Let $G = (N,E)$ be a directed graph.
Question: Is there a nontrivial semikernel ($SK \neq \emptyset$) for $G$?

The next theorem states that this problem is intractable.

**Theorem 5.1.** Determining whether a graph has a nontrivial semikernel is NP-complete.

**Proof.** The proof is by reduction from 3-SAT. Given a formula in CNF $C = \{C_1, C_2, \ldots, C_n\}$, $C_i = C_{i1} \lor C_{i2} \lor C_{i3}$ we construct a graph $G = (N,E)$, shown in Fig. 2, as follows: For every literal $x_i$ (and its negation) we put a node $n_i$ ($n'_i$ respectively) in the set $N$. We refer to this set of nodes as $L$. For every clause $C_i$ in $C$ put a node $c_i$ in $N$ (we call this set of nodes $S$), as well as a node $Aux$ and a cycle
of length 3 involving a distinguished node $A$. The set $E$ consists of the following edges:

1. There exists a bidirectional edge between the nodes which correspond to complementary literals,
2. For every literal $x_i$ occurring in clause $C_j$ there is an edge $(n_i, c_j)$,
3. From every $c_i \in S$ there is an edge $(c_i, Aux)$,
4. There exists the edge $(Aux, A)$, as well as an edge $(A, n_i)$ from $A$ to every node $n_i \in L$.

**Lemma 5.2.** In graph $G$ every nontrivial semikernel (if one exists) is a kernel.

**Proof.** Let $SK$ be a semikernel of $G$, $SK \neq \emptyset$. Assume that $SK$ contains a nonempty set of nodes $M \subseteq S$. Then the node $Aux$ does not belong to $SK$, and $A$ can not belong to $SK$ as well (note that $A$ is one of the nodes in the odd cycle). Since every node in $M$ receives an edge from some node in $L$, the set $\Gamma^-(M)$ must be covered. Assume that some of the nodes in $L$ belong to $SK$ and cover them. But all of these nodes receive an edge from $A$, and $A$ cannot be covered by $SK$. Hence no semikernel can contain a node from $S$. Assume now that there is a semikernel $SK \neq \emptyset$ such that $Aux \notin SK$. Since $A$ cannot be covered, none of the nodes of $L$ can belong to $SK$. Since none of the nodes in $S$ can belong to $SK$ as well then $SK$ is empty. Hence, every nontrivial semikernel must contain $Aux$. Therefore, for every semikernel $SK$, $S \in \Gamma^+(SK)$ must hold. For this to happen a subset of the nodes of $L$ that covers all nodes in $S$ must be in $SK$. It is easy to see that such a semikernel is kernel.

Hence in $G$ every nontrivial semikernel is a kernel. $\square$

Observe that every nontrivial semikernel (if one exists) implies a satisfying truth assignment to the literals of the clauses, and vice versa. Hence every polynomial algorithm for this decision problem would also solve the 3-SAT problem in polynomial time. $\square$

**Proposition 5.3.** The decision problem whether a negative logic program has a well-founded interpretation (or a partial stable model) different from the empty set is NP-complete.

**Proof (sketch).** Let $G = (N, E)$ be a graph. We can construct an associated negative logic program $P_G$ as follows: for every $n_i \in N$ add to $P_G$ the atom $p_i$ and the rule $r_i : p_i \leftarrow \neg p_1, \neg p_2, \ldots, \neg p_m$, where for every $1 \leq j \leq m$, $(n_j, n_i) \in E$ holds.

Let $I$ be a well-founded interpretation for $P$. Define $K = \{n_i|body(r_i) \in I^-\}$. We will show that $K$ is a semikernel for $G_P$. Since $I$ is self-consistent $K$ is an independent set. We now show that $K$ is semidominant. Let $n_j \in \Gamma^-(n_i)$, $n_i \in K$. Then $\neg p_j \in body(r_i)$, hence $\neg p_j \in I^-$. By $I^- \subseteq \forall(I^-)$ we have that there is a rule $n_k \in K$ such that $(n_k, n_j) \in E$. 

Let now $K$ be a semikernel for $G_p$ and define $I = I^- \cup (I^-)^\ast$, where $I^- = \{-p_j | -p_j \in body(n_i), n_i \in K\}$. Since $K$ is independent $I$ is self-consistent. We prove now that $I^- \subseteq \mathcal{U}(I^-)$. Let $-p_j \in I^-$. Then either there is no rule with $p_j$ in its head or for every such rule $r_j$, it holds that $r_j \in \Gamma^+(K)$ which means that $-p_j \in \mathcal{U}(I^-)$. □

We show next how an algorithm that computes kernels, can also be used for computing semikernels.

**Definition 5.4.** A graph $G' = (N', E')$ is the semikernel equivalent of a graph $G = (N, E)$, if $N' = N \cup \{n' | n \in N\}$, and $E' = E \cup \{(n, n'), (n', n) | n \in N, n' \in N' - N\} \cup \{(n_i, n'_i) | n_i \in N, n'_i \in N', \text{ and } n_j \in \Gamma^+_C(n_i)\}$.

**Theorem 5.5.** If $G'$ is the semikernel equivalent to $G$, then, if $K$ is a kernel for $G'$, $K - N'$ is a semikernel for $G$. Conversely, every semikernel in $G$ induces a kernel in $G'$.

**Proof.** Let $K$ be a kernel for $G'$. The nodes of $K - N'$ are independent and for every node $n_j$ such that $n_j \in \Gamma^+_C(n_i)$ and $n_i \in K - N'$, since $(n'_j, n_i) \in E$, there must be a node $n_k \in K - N'$ such that $(n_k, n_j) \in E$. Hence $K - N'$ is a semikernel for $G$.

Let $K$ be a semikernel for $G$. Then see that $K \cup \{n | n \in N' - N, n \notin \Gamma^+(K)\}$ is a kernel for $G'$.

In view of the above theorem, every algorithm that computes the kernels of a graph $G$ is also capable of computing the semikernels of $G$ if it is supplied with the semikernel equivalent of $G$.

In the following we briefly present some other results obtained using the graph theoretic transformation of logic programs.\(^\text{13}\) Clearly in the case of graphs without cycles the computation of the unique kernel, which coincides with the maximal semikernel, is trivial. The kernel in this case captures, what most researchers agree to be, the meaning of the associated logic program.

In cases of graphs with cycles there are two possibilities. The first is that the graph is odd-cycle free, and a kernel always exists. In this case we can perform the tie-breaking procedure introduced in [31] and compute nondeterministically in polynomial time a kernel of the graph. Furthermore, as shown in [14], there is a polynomial delay algorithm\(^\text{14}\) which enumerates a set of kernels for this class of graphs, called "standard" kernels. Unfortunately this procedure is not complete, that is, there may be kernels that can not be detected by the procedure. Nevertheless it may serve as a sound but incomplete reasoning method for logic programs without odd cycles. It is also shown in [14] that determining whether there are other kernels for such a

\(^{13}\) For a survey of complexity results regarding logic programs refer to [30, 10].

\(^{14}\) We say that an algorithm for generating configurations is polynomial delay [25] if there is only a polynomial delay between any two configurations generated. Such algorithms may behave exponentially because of the exponentially many different configurations, but this is obviously unavoidable.
graph, different from those found by this procedure, is NP-complete. Finally skeptical reasoning with these graphs (i.e. whether a node is contained in every kernel) was also proved to be intractable in [13].

In the case of graphs with odd cycles there is still a possibility to maintain the above computational features at the cost of incompleteness. Suppose that the task is to compute the kernels of a graph. First remove from the graph at hand the edges (or nodes) causing the odd cycles. Compute the "standard" kernels for the new odd-cycle free graph with polynomial delay. Determine which of these kernels are kernels for the original graph. The overall complexity is bounded by the size of the graph and the number of "standard" kernels in the odd-cycle free graph. The procedure is sound but incomplete.

We also note that if the graph is symmetric, then a set K is a kernel iff K is a maximal independent set (MIS) for the undirected graph obtained after removing the direction from the edges in the original graph. Every graph has at least one MIS which can be computed in polynomial time. Furthermore there are polynomial delay procedures that compute all the MIS of a graph (see [44], [25]).

Given the intractability of computing kernels and semikernels in the general case, another possibility is to look for approximations to these problems. The major obstacle is that it is not easy to find a measure for the approximation. Some recent attempts include the approximate entailment of [9] referring to default logic and circumscription, which extends their previous work in [8] regarding classical logic. The graph-theoretic representation offers yet another possibility. Namely, by approximations we mean the efficient and (in cases where this is necessary) nondeterministic computation of subsets of the maximal semikernels of the graphs, which are, themselves, semikernels. According to this view, given a graph G, a semikernel S1 is a better approximation than a semikernel S2, if S2 ⊆ S1, where S1, S2 ⊆ S and S is a maximal semikernel for G.

6. The case of default theories

Since Reiter’s original definition of default logic, several researchers have given different definitions especially to the notion of the extension,\(^{15}\) as well as to the notion of the default rule (e.g. [22]). Most of these proposals intent to rebut some of the original default logic shortcomings (e.g. the nonexistence of extensions, difficulties in expressing disjunctive information, etc.). On the other hand, the various semantics for logic programs have been applied to default logic and other nonmonotonic formalisms. An early attempt in this direction is described in [34] where the well-founded semantics is defined for default and autoepistemic theories, based on a three-valued reconstruction of these formalisms. More recently well-founded semantics for the same formalisms has been proposed in [1, 2] based on an ordering for the sets of interpretations around which the Gelfond–Lifschitz operator oscillates. In [32] another reformulation of default logic

\(^{15}\) See [17] for a general framework where several invariants of default logic are examined.
is presented that satisfies some criteria defined by the authors and is along the lines of stable models for extended logic programs. In [35] the stationary extensions are presented, an extension of the stationary semantics for logic programs. The approaches in [26, 6] are closely related to the framework developed in this section. Finally, in a recent book [30], several definitions of the notion of the extension are given. In Section 6.2 we compare some of these approaches with the framework we present in the following.

6.1. Default logic semantics

In this section we present a framework for reasoning with propositional default theories. It is not our intention to give a completely new semantics but rather to generalize the semantic for logic programs studied in the previous sections. In contrast to most of the approaches to default logic mentioned above, we give several quasi-inductive definitions of the notion of extension. These notions capture different methods (or modes) of reasoning with default logic and link the graph theoretic results presented earlier to the case of conjunctive default theories.

Throughout this section we refer exclusively to seminormal propositional default theories, except if otherwise stated. In a seminormal default theory every default is of the form \(A : MB \land C/C\), where \(B \land C\) is consistent. Furthermore, the basic theory refers to the case of conjunctive default theories where \(W\), the prerequisite, the justification and the consequent of the rules are conjunctions (sets) of literals. We drop this restriction later, and generalize the notions to the general case of propositional default theories. A conjunctive theory that contains defaults of the form \(a : \bigwedge b_i/Mb, A\), where each \(b_i\) is a consistent conjunction, is a seminormal theory if for every default \(w \subseteq b_1 \cup \cdots \cup b_n\) holds.

Let \(A = (D, W)\) be a conjunctive default theory. Any literal from the language of \(\Delta\) can be considered as an assumption. A set of assumptions is called a hypothesis. Intuitively a hypothesis is a set of literals assumed consistent with the semantics of the theory. A hypothesis may contain both a literal and its negation. We denote by \(D_H\), where \(H\) is a hypothesis, the set of defaults obtained by deleting from the set \(\text{Just}(d_i)\) of every rule \(d_i \in D\), the justifications in \(H\), and then deleting every rule \(d_j \in D\), such that \(\text{Just}(d_j) \neq \emptyset\).

A hypothesis \(H\) supports a literal \(\alpha\) in \(\Delta\) (denoted by \(H \xrightarrow{\Delta} \alpha\)) if there is a sequence of defaults \(d_1, \ldots, d_k \in D_H\) such that \(\text{Prer}(d_i) \subseteq \text{Th}(\bigcup_{j=1}^{i-1} \text{Cons}(d_j) \cup W)\), \(1 \leq i \leq k\), and \(\alpha \in \text{Th}(\text{Cons}(d_1) \cup \cdots \cup \text{Cons}(d_k) \cup W)\).

A hypothesis \(H\) attacks another hypothesis \(H'\) in a theory \(\Delta\) (denoted by \(H \xrightarrow{\Delta} H'\)) if \(H \xrightarrow{\Delta} \beta\) for some \(\neg \beta \in H'\). A hypothesis is self-consistent if it does not attack itself. An assumption \(\beta\) is unfounded with respect to a hypothesis \(H\) if for every \(H'\) such that \(H' \rightarrow \beta\) we have \(H \rightarrow H'\). We denote by \(\mathcal{U}_A(H)\) the set of all unfounded assumptions w.r.t. \(H\) in a theory \(\Delta\). We say that a set of propositions \(E\) is supported if there is a hypothesis \(H\) that supports every proposition in \(E\). If \(E\) is the deductive closure of a set of propositions supported in \(\Delta\) by a hypothesis \(H\) such that \(\neg H = \mathcal{U}_A(H)\), and \(H\) is self-consistent, then \(E\) is a partial extension for \(\Delta\).
It is important to notice the relation between partial extensions and the semantics of logic programs. If $P$ is a logic program we denote by $\text{tr}(P)$ the default theory $(D, W)$ where $W = \emptyset$ and $D$ contains a rule $a_1 \land \cdots \land a_n : M \neg b_1, \cdots, M \neg b_k / c$ for every rule $c \leftarrow a_1, \ldots, a_n, \neg b_1, \ldots, \neg b_k$ in $P$. The next theorem demonstrates that there is a direct correspondence between the partial extensions of $\text{tr}(P)$ and the complete well-founded interpretations of $P$.

**Theorem 6.1.** Let $P$ be a logic program. Then $E = H \cup H^p_{\text{tr}(P)}$ is a complete well-founded interpretation of $P$ iff $\text{Th}(H^p_{\text{tr}(P)})$ is a partial extension of $\text{tr}(P)$.

**Proof (sketch).** Let $E = H \cup H^p_{\text{tr}(P)}$ be a complete well-founded interpretation of $P$. We will show that $\text{Th}(H^p_{\text{tr}(P)})$ is a partial extension of $\text{tr}(P)$. First note that $H \models P \alpha$ for some atom $\alpha$ iff $H \models P_{\text{tr}(P)} \alpha$. Hence, $H \models P H'$ iff $H \models P_{\text{tr}(P)} H'$. Consequently, $\neg b \in \mathcal{U}_P(H)$ iff $b \in \mathcal{U}_P(H)$. Since $H$ is complete for $P$, $H = \mathcal{U}_P(H)$ and therefore $\neg H = \mathcal{U}_{\text{tr}(P)}(H)$. Moreover, $H$ is self-consistent. Finally, $H^p_{\text{tr}(P)}$ contains the maximal set of atoms supported in $P$ by $H$, hence the same holds for $\text{tr}(P)$. Therefore, $\text{Th}(H^p_{\text{tr}(P)})$ is a partial extension of $\text{tr}(P)$.

Assume now that $E'$ is a partial extension for $\text{tr}(P)$ supported by a hypothesis $H$. Let $E$ be the set of atoms in $E'$. Since $\text{tr}(P)$ is a conjunctive theory, $E' = \text{Th}(E)$. Then $E$ contains all the atoms supported by $H$ in $\text{tr}(P)$, hence $E = H^p_{\text{tr}(P)}$. Note that for every atom $\alpha$, $H \models P \alpha$ iff $H \models P_{\text{tr}(P)} \alpha$. Hence, $b \in \mathcal{U}_P(H)$ iff $b \in \mathcal{U}_P(H)$. Therefore, since $\neg H = \mathcal{U}_{\text{tr}(P)}(H)$ holds for $\text{tr}(P)$, $H = \mathcal{U}_P(H)$ holds for $P$. Finally, since $E$ is the maximal set of atoms supported by $H$ in $\text{tr}(P)$ and $H$ is self-consistent wrt $\text{tr}(P)$, it follows that $H \cup E = H \cup H^p_{\text{tr}(P)} = H \cup H^p_{\text{tr}(P)}$ is a complete well-founded interpretation of $P$. \qed

The next theorem provides a proof-theoretic definition for the partial extensions in the vein of [36].

**Theorem 6.2.** A set of propositions $E$ is a partial extension for a propositional semi-normal conjunctive default theory $\Delta = (D, W)$ iff $E = \bigcup_{i=0}^{\infty} E_i$ for a sequence of sets $E_i$ such that

$$F_0 = W \text{ and }$$

$$\forall i, i > 0, E_i = \text{Th}(E_{i-1}) \cup \{w|a : Mb_1, \ldots, Mb_n / w, a \in E_{i-1}, \text{ and for every } b_i, \neg b_i \notin E \text{ and } \neg b_i \in B_E\},$$

where

$$B_E = \{p| p \text{ is a literal, } p \notin W \text{ and for every sequence of defaults } d_1, \ldots, d_n \in D, \text{ such that } p \in \text{Cons}(d_n), \text{ and for every } d_i \text{ in the sequence, } \text{Prer}(d_i) \subseteq \bigcup_{j=1}^{i-1} \text{Cons}(d_j) \cup W, \text{ the condition } \exists q \in \text{Just}(d_i) \text{ such that } \neg q \in E \text{ holds } \}.$$
Proof (sketch). We first show that if \( E \) is a partial extension supported by a hypothesis \( H \), then there is a sequence of defaults satisfying the conditions of the theorem, leading to \( E \). First we prove that \( \mathcal{U}_A(H) = B_E \). If \( b \in \mathcal{U}_A(H) \) then either there is no sequence of defaults that proves \( b \), or for every such sequence there is a proposition \( p_i \in \text{Just}(d_i) \) for some \( d_i \) in the sequence, such that \( \neg p_i \in E \). Then see that \( b \in B_E \). Similarly if \( b \notin \mathcal{U}_A(H) \) then \( b \notin B_E \). Hence \( \mathcal{U}_A(H) = B_E \).

For every \( q \in E \) there is a sequence of defaults \( d_1, \ldots, d_k \) that proves \( q \), such that \( \text{Prer}(d_i) \subseteq \bigcup_{j=1}^{i-1} \text{Cons}(d_j) \), 1 \( \leq i \leq k \), and for every \( p \in \text{Just}(d_i) \) of every \( d_i \) in the sequence, \( p \in H \) holds. Since \( \mathcal{U}_A(H) = B_E \) and \( \mathcal{U}_A(H) = \neg H \) then \( \neg p \notin B_E \). Moreover, for every \( p \in \text{Just}(d_i) \) of every \( d_i \) in the sequence \( \neg p \notin E \) holds, because otherwise \( H \) is not self-consistent. Hence every \( q \in E \) satisfies the conditions of the theorem. On the other hand if \( q \notin E \) then \( q \) is not supported by \( H \). This means that for every sequence of rules proving \( q \) (if any) there is \( p_i \in \text{Just}(d_i) \) for some \( d_i \) in the sequence, such that \( p_i \notin H \) and since \( B_E = \neg H \), \( \neg p_i \notin B_E \), and we are done.

We show now that if \( E \) is a set satisfying the conditions of the theorem then it is a partial extension. First see that \( E \) is supported by the set \( \neg B_E \). We prove that \( \mathcal{U}_A(\neg B_E) = B_E \). Assume that \( b \in \mathcal{U}_A(\neg B_E) \). Then either there is no sequence of defaults that proves \( b \), or every such sequence is blocked by \( \neg B_E \). Then \( b \in B_E \). Conversely if \( b \in B_E \) then again, every sequence of defaults proving \( b \) (if one exists) is blocked by some \( c \in E \), or in other words, there is an attack on \( b \) by a set of literals in \( \neg B_E \) which is the union of the justifications of the rules that prove \( b \). Hence \( b \in \mathcal{U}_A(\neg B_E) \), and \( \mathcal{U}_A(\neg B_E) = B_E \). Moreover, \( E \) is the maximal set of propositions supported by \( \neg B_E \). Assume now that \( \neg B_E \) is not self-consistent. Then there exists a literal \( b \) such that \( b \in \neg B_E \) and \( \neg b \in E \). Then \( \neg b \in B_E \) and \( \neg b \in E \). The first condition, \( \neg b \in B_E \), means that for every sequence of defaults that prove \( \neg b \) there is a literal \( m \) in the justification of some default of the sequence such \( \neg m \in E \). On the other hand, \( \neg b \in E \), means that there exists a sequence of defaults that prove \( \neg b \), such that for every justification \( p \) in the sequence \( \neg p \notin E \) holds. This contradicts the first condition and therefore \( E \) is self-consistent. Hence, \( E \) is a partial extension.  

Roughly speaking, a rule can be applied only if every rule which provides the negation of a justification of the rule is blocked by the activated rules. The consistency of the justifications with the partial extension is necessary in order to avoid inconsistent extensions. Consider, for example, the theory \( W = \emptyset \) and \( D = \{ M\bar{B}/B,M\bar{B}/\neg B \} \). If we do not require the consistency of the assumptions with the extensions, then the inconsistent set \( E = \{ B, \neg B, \ldots \} \) is a partial extension, since \( B_E = \{ B, \neg B, \ldots \} \).

We define the semantics of the default theory to be its maximal partial extensions. As in the case of logic programs, for every conjunctive default theory there is a transformation, that preserves partial extensions, to a conjunctive prerequisite-free default theory.
Theorem 6.3. Let \( \Lambda = (D, W) \) be a conjunctive default theory and \( \Lambda^- = (D^-, W) \) be the prerequisite free conjunctive default theory that contains for every minimal support of a literal \( \alpha \in \text{Cons}(r) \), \( r \in D \), a rule \( r' \in D^- \) with \( \text{Cons}(r') = \{ \alpha \} \) and \( \text{Just}(r') \) a minimal support of \( \alpha \). Then \( E \) is a partial extension for \( \Lambda \) iff \( E \) is a partial extension for \( \Lambda^- \).

Proof (sketch). Let \( E \) be a partial extension of \( \Lambda = (D, W) \). We show that \( E \) is also an extension for \( \Lambda^- = (D^-, W) \), and furthermore \( B_{E,\Lambda} = B_{E,\Lambda^-} \). We first show the second equality. Assume that for some literal \( p, p \in B_{E,\Lambda} \) holds. There are two reasons for this to happen. First there is no sequence of defaults \( d_1, \ldots, d_n \in D \), such that \( p \in \text{Cons}(d_n) \), and for every \( d_i \) in the sequence, \( \text{Prer}(d_i) \subseteq \bigcup_{j=i}^{n-1} \text{Cons}(d_j) \cup W \). See that in this case there is no support for \( p \) and there will be no rule in \( D^- \) with head \( p \) hence \( p \in B_{E,\Lambda^-} \). The second possibility is that for every such sequence of defaults, there exists a \( q \in E, \neg q \in \text{Just}(d_i) \) for some rule \( d_i \) in the sequence. But then every rule in \( \Lambda^- \) concluding \( p \) will also contain such a proposition, hence \( p \in B_{E,\Lambda^-} \). Similar arguments can be constructed for the opposite direction, hence \( B_{E,\Lambda} = B_{E,\Lambda^-} \).

Now assume that \( b \in E_{\Delta} \). Then there is a minimal sequence of defaults \( d_1, \ldots, d_n \in D \), which can lead to the derivation of \( b \) from \( W \) and if \( p \in \text{Just}(d_i) \cup \ldots \cup \text{Just}(d_n) \) then \( \neg p \in B_{E,\Delta} \). But then there will be a rule \( d_k \in D^- \), \( \text{Just}(d_k) = \text{Just}(d_1) \cup \ldots \cup \text{Just}(d_n) \), with \( b \in \text{Cons}(d_k) \). Since \( B_{E,\Lambda} = B_{E,\Lambda^-} \), \( b \in E_{\Delta^-} \) also holds. If \( b \notin E_{\Delta} \), then for every sequence of rules (if one exists) that proves \( b \), there is \( p \in \text{Just}(d_i) \) for some \( d_i \) in the sequence, such that \( \neg p \notin B_{E,\Delta} \). This means that \( b \notin E_{\Delta^-} \) as well. Hence if \( E \) is an extension of \( \Lambda = (D, W) \) then \( E \) is an extension of \( \Lambda^- = (D^-, W) \).

We now show the opposite direction. Assume that \( E \) is an extension for the theory \( \Lambda^- \). First, using similar to the above given arguments, we can prove again that \( B_{E,\Lambda} = B_{E,\Lambda^-} \). If \( b \in E_{\Delta^-} \) for some literal \( b \), then this means that there is a sequence of defaults in \( \Lambda, d_1, \ldots, d_k \in D \) such that \( b \in \text{Cons}(d_k) \) and for every \( p \in \text{Just}(d_1) \cup \ldots \cup \text{Just}(d_k) \), \( \neg p \notin B_{E,\Delta} \). Furthermore, if \( q \in \text{Prer}(d_i) \) for some \( d_i \) in the sequence, then \( q \in \text{Cons}(d_j) \), \( j < i \) or \( q \in W \). Hence \( b \in E_{\Delta} \) as well. Finally assume that \( b \notin E_{\Delta} \). This means either that there is no rule in \( D^- \) with \( b \) as its consequent or for every such rule there is a proposition the negation of which does not belong to \( B_{E,\Delta^-} \). Then, \( b \notin E_{\Delta} \) holds as well. \( \square \)

Notice that during the translation we may need to distribute the literals of the justification over several \( M \) operators, in order to avoid inconsistent justifications. Consider for example

\[
W = \{ A \} \quad \text{and} \quad D = \left\{ \frac{A : MB}{B}, \frac{A : M \neg B \land C}{C}, \frac{B \land C : MD}{D} \right\}.
\]

The associated prerequisite free theory is

\[
W = \{ A \} \quad \text{and} \quad D' = \left\{ \frac{:\ MB}{B}, \frac{: M \neg B \land C}{C}, \frac{MB, M \neg B, MC \land D}{D} \right\}.
\]
In a way similar to logic programs, we define now the *rule graph* of a prerequisite-free conjunctive default theory.

**Definition 6.4.** Let $\Delta$ be a prerequisite-free conjunctive default theory. The *rule graph* of $\Delta$, denoted by $\mathcal{R}(\Delta)$, is the directed graph $(\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{r : r \in \Delta\}$ and $\mathcal{E} = \{(r_k, r_m) : \text{if } b \in \text{Cons}(r_k) \text{ and } \neg b \in \text{Just}(r_m)\}$.

In [13], it has been proved that the Reiter extensions of a prerequisite-free conjunctive default theory correspond to the kernels of the theory's rule graph. The next two theorems refer to the relation between the partial extensions of a theory and the semikernels of its associated rule graph.

**Theorem 6.5.** Let $\Delta$ be a prerequisite-free conjunctive default theory and $\mathcal{R}(\Delta) = (\mathcal{V}, \mathcal{E})$ its rule graph. If $E$ is a partial extension of $\Delta$, then for the set $S = \{r : r \in \mathcal{V}, \text{Just}(r) \subseteq \neg B_E\}$, $S = S_1 \cup S_2$ holds, where $S_1$ is a semikernel of $\mathcal{R}(\Delta)$, and $S_2$ is the initial acyclic part for the graph $\mathcal{R}(\Delta)/(S \cup \Gamma^+(S_1))$.

**Proof (sketch).** Let $E$ be a partial extension for the prerequisite-free conjunctive default theory $\Delta$. We will show that for $S = \{r : \text{Just}(r) \subseteq \neg B_E\}$, $S = S_1 \cup S_2$ holds, for $S_1, S_2$ as described above. Since any of the sets $S_1, S_2$ can be empty and $S_1 \cup S_2$ is always a semikernel, it suffices to show that $S$ is a semikernel for $\mathcal{R}(\Delta)$, and the initial acyclic part of $\mathcal{R}(\Delta)/(S \cup \Gamma^+(S))$ is empty. If a node $r_i \in S$ receives an incoming edge from some other node $r_j$ in $\mathcal{R}(\Delta)$, this means that there is a literal in $\text{Just}(r_i)$ the negation of which is in the consequents of $r_j$. But since the negation of the literal belongs to $B_E$ this means that there must be some node $r_k \in S$, for which $(r_k, r_j) \in \mathcal{E}$ holds. Hence $S$ dominates all the nodes which belong to $\Gamma^-(S)$. Moreover, since $E$ is consistent, $S$ is an independent set. On the other hand since for every rule $m \not\in S \cup \Gamma^+(S)$ there is always a proposition $p \in \text{Just}(m)$, $p \not\in \neg B_E$, node $m$ receives edges from some nodes in $\mathcal{R}(\Delta)/(S \cup \Gamma^+(S))$. Hence the initial acyclic part of $\mathcal{R}(\Delta)/(S \cup \Gamma^+(S))$ is empty.

**Theorem 6.6.** Let $\Delta = (D, W)$ be a prerequisite-free conjunctive default theory, $\mathcal{R}(\Delta)$ its rule graph, $K$ a semikernel of $\mathcal{R}(\Delta)$, and $IP$ the initial acyclic part of $\mathcal{R}(\Delta) - (K \cup \Gamma^+(K))$. Then $E = \text{Th}\{p : p \in \text{Cons}(d_i) \text{ and for the associated with } d_i \text{ node } n_i \in \mathcal{R}(\Delta), n_i \in K \cup IP \text{ holds}\}$, is a partial extension for $\Delta$.

**Proof. (sketch).** The partial extension $E$ is generated by the set of defaults $D'$, such that for every $d_i \in D'$, its associated node $n_i$ belongs to $K \cup IP$. Define $B_E = \{p : \exists d \in D \text{ such that } p \in \text{Cons}(d) \text{ or } \forall r \in D \text{ such that } p \in \text{Cons}(d), \text{ for the associated with } d \text{ node } n, n \in \Gamma^+(K) \cup \Gamma^+(IP) \text{ holds}\}$. Then, for every $p \in \text{Just}(d_i)$ and $d_i \in D'$, $\neg p \in B_E$ holds. Furthermore, for every $p \in \text{Just}(d_i)$ and $d_i \in D'$, $\neg p \not\in E$ holds, because otherwise $K \cup IP$ is not a semikernel.
Example 6.7. Consider the theory $\Delta$ of Example 6.13. We first convert the theory into one without prerequisites, obtaining in this way the theory $\Delta' = (D', W)$, where

$$D' = \left\{ \frac{MB}{B}, \frac{M \neg B}{\neg B}, \frac{MC \land \neg B}{C}, \frac{ME \land \neg G \land \neg C}{E}, \frac{MF \land \neg E}{F}, \frac{MG \land \neg F}{G} \right\}.$$ 

The rule graph of this theory is depicted in Fig. 3.

The graph has two (maximal) semikernels $K_1 = \{r_2, r_3, r_5\}$ and $K_2 = \{r_1\}$ which correspond to the two maximal partial extensions. The first is a kernel, hence, the associated partial extension is also a (Reiter) extension.

Theorem 6.2 allows us to extend the definition of the partial extensions to the general case of default theories, in a straightforward manner.

Definition 6.8. We define a set $E$ to be a partial extension for a default theory $\Delta = (D, W)$ iff $E = \bigcup_{i=0}^{\infty} E_i$ for a sequence of sets $E_i$ such that

$$E_0 = W \text{ and }$$

$$\forall i, i > 0, \ E_i = \text{Th}(E_{i-1}) \cup \{ w | a : Mb_1 \ldots Mb_n / w, a \in E_{i-1} \text{ and for every } b_i, \ 1 \leq i \leq n, b_i = \bigwedge b_{ij} \text{ where } b_{ij} \text{ is disjunction of literals}, \ b_i \text{ is consistent with } E \text{ and } \neg b_{ij} \in B_E \}.$$

where $B_E = \{ p | \text{ if for every sequence of defaults } d_1, \ldots, d_n \in D, n \geq 0, \text{ such that } \text{Th}(W \cup \bigcup_{j=1}^{n} \text{Cons}(d_j)) \models p, \text{ and for every default } d_i \text{ in the sequence } \text{Prer}(d_i) \subseteq \text{Th}(W \cup \bigcup_{j=1}^{i-1} \text{Prer}(d_j)), \text{ the condition } \exists q \in \text{Just}(d_i) \text{ such that } \neg q \in E \text{ holds } \}$. 

The notion of partial extension captures a credulous form of reasoning with default theories. However, there are cases where skeptical reasoning is more appropriate. Skepticism in default reasoning can be captured by the deterministic and well-founded extensions, which are notions developed earlier for logic programs.
Definition 6.9. Let $A$ be a default theory and let $E$ be a set of propositions. We say that $E$ is a

Deterministic (partial) extension: if $E$ is a partial extension contained in every maximal partial extension.

Well-founded (partial) extension: if $E$ is a minimal deterministic extension.

For the case of well-founded extensions the next theorem introduces a quasi-inductive characterization, for general propositional theories.

Theorem 6.10. A set $E$ is a well-founded extension for a default theory $A = (D, W)$ iff $E = \bigcup_{i=0}^{\infty} E_i$ for a sequence of sets $E_i$ such that

\[ E_0 = W \quad \text{and} \quad E_i = \text{Th}(E_{i-1}) \cup \{ w | a : M b_1 \ldots M b_n / a, a \in E_{i-1} \}
\]

and for every $b_i, 1 \leq i \leq n$, $b_i = \bigwedge b_{ij}$ where

$b_{ij}$ is a disjunction of literals, $b_i$ is consistent with $E$ and $\neg b_{ij} \in B_E$, where

$B_E = \{ p \mid \text{for every sequence of defaults } d_1, \ldots, d_n \in D \text{ such that } \text{Th}(W \cup \bigcup_{j=1}^{n} \text{Cons}(d_j)) \vdash p, \text{and for every default } d_i \text{ in the sequence } \text{Prer}(d_i) \subseteq \text{Th}(W \cup \bigcup_{j=1}^{i-1} \text{Prer}(d_j)), \text{the condition } \exists q \in \text{Just}(d_i) \text{ such that } \neg q \in \text{Th}(E_{i-1}) \text{ holds} \}$.

Proof (sketch). First see that the set $E$ defined above is a deterministic extension. Furthermore any smaller set is not a partial extension, hence $E$ is the minimal deterministic model.

On the other hand the well-founded model can be expressed as a sequence of sets $E_0, E_1, \ldots$ as defined above. \(\square\)

Notice the occurrence of $E$ in the definition of $E$ itself. This is necessary for the general case of theories, but in the case of seminormal theories the consistency check of the justification with the well founded extension is redundant. For instance, consider the theory $W = \{ A \}$ and $D = \{ A : M D / B, A : M F / \neg B \}$. If we omit the consistency check of the justifications wrt $E$ then we get $WFE_1 = \{ B, \neg B, \ldots \}$, while with the consistency check we obtain $WFE_2 = \{ A \}$. Thus, in the case of seminormal defaults, the definition is constructive and deterministic. As a consequence the well-founded extension of a theory is unique.

Example 6.11. Let $A = (D, W)$, where

\[ W = \{ A \} \quad \text{and} \quad D = \left\{ \begin{array}{c}
A : M \neg K \land B \\
B \\
A : M \neg C \land \neg B \\
B : MC \land \neg D \\
C : MD \land \neg F \\
C : MF \land E \land G \\
C : M \neg E \land \neg G \\
C : ME \land \neg G \land H \\
C : M \neg H \\
E \land G \\
\neg E \land \neg G \\
H \\
\neg H
\end{array} \right\}.
\]
Theory \( A \) has two maximal partial extensions (which are also Reiter extensions) namely, \( E_1 = \{ A, B, C, \neg D, E, G, \neg H \} \) and \( E_2 = \{ A, B, C, \neg D, \neg E, \neg G, \neg H \} \). The deterministic extensions are \( DE_1 = \{ A, B, C, \neg D \} \), \( DE_2 = \{ A, B, C, \neg D, \neg H \} \). The well-founded extension of \( A \) is the set \( DE_1 \).

### 6.2. Comparison with other approaches

In this section we discuss the relation of our approach to some other formalizations of default logic. This is done by means of a set of examples which show that our semantics is different from all the main approaches to default logic. Detailed discussion of the relation of the framework introduced here to the other formalisms will be the subject of a subsequent paper.

We start by comparing our semantics to Reiter extensions. It is easy to show that every Reiter extension is a maximal partial extension.

**Theorem 6.12.** Let \( A \) be a default theory and \( E \) a Reiter extension of \( A \). Then \( E \) is also a maximal partial extension for \( A \).

**Proof (sketch).** Let \( GD(E, A) \) be the set of generating defaults of \( E \). Then every justification of the defaults of \( GD(E, A) \) will be consistent with \( E \).

Assume that \( b_i \) belongs to the justifications of a rule \( r_i \in GD(E, A) \) and \( \neg b_i \notin B_E \). Then there is a rule with \( \neg b_i \) in its consequents that is applicable wrt to \( E \). Since \( E \) is closed under the rules of \( D \), \( \neg b_i \in E \). But then \( E \) is not an extension. Hence for every \( b_i \) that belong to the justifications of some rule \( r_i \in GD(E, A) \), \( \neg b_i \in B_E \) must hold.

Finally for every default \( d \) in \( D - GD(E, A) \), there must be \( b \in \text{just}(d) \) such that \( \neg b \notin E \), and therefore \( \neg b \notin B_E \). Hence, according to Theorem 6.2, \( E \) is a partial extension. It is easy to see that \( E \) is also a maximal partial extension.

However it is not the case that every (maximal) partial extension is a Reiter extension, as demonstrated by the following example.

**Example 6.13.** Let \( A = (D, W) \) be a default theory, where

\[
W = \{ A \} \quad \text{and} \quad D = \left\{ \begin{array}{c}
A : MB, \ A : M \neg B, \ A : MC \land \neg B, \\
B, \neg B, \quad C, \neg C, \quad E, \neg E, \quad F, \neg F, \quad G, \neg G
\end{array} \right\}.
\]

Theory \( A \) has two maximal partial extension, namely \( E_1 = \text{Th}(\{ A, B, C, E \}) \) and \( E_2 = \text{Th}(\{ A, B \}) \). Notice that \( E_1 \) is also a Reiter extension, while \( E_2 \) is not, due to the presence of the last three rules.

We now consider the semantics for default logic introduced in [30]. The weak extensions are similar to Reiter extensions, except that instead of proving the prerequisites of the rules, we can assume them to hold in the extension (similarly to autoepistemic
stable expansions). In fact, weak extensions are quite different from partial extensions. Consider the following example.

**Example 6.14 (Weak extensions, Marek and Truszczynski [30]).** Let $\Delta$ be the theory $\Delta = (D, W)$ with

$$W = \emptyset \quad \text{and} \quad D = \left\{ \frac{M \neg A}{D}, \frac{M \neg C}{A}, \frac{M \neg A}{B}, \frac{M \neg B}{C} \right\}.$$ 

While there is no weak extension (as well as Reiter extension) the partial extension of $\Delta$ is $E = \text{Th}(\{D\})$.

Partial extensions are different from minimal sets since partial extensions need to be neither minimal nor closed under the defaults. Furthermore, a minimal set may become closed by adding to it the negation of a justification. This operation is not allowed for partial extensions.

**Example 6.15 (minimal sets, Marek and Truszczynski [30]).** Let

$$W = \emptyset \quad \text{and} \quad D = \left\{ \frac{M \neg A}{B} \right\}.$$ 

There is only one partial extension $E = \text{Th}(\{B\})$, but there are two minimal sets, $E$ and $E' = \text{Th}(\{A\})$.

Now let

$$W = \{A \rightarrow B\} \quad \text{and} \quad D = \left\{ \frac{B}{A}, \frac{M \neg B}{A} \right\}.$$ 

The partial extension of this theory is $E = \text{Th}(\{A \rightarrow B\})$ and its minimal set (which is also a weak extension) is $E' = \text{Th}(\{A, B\})$.

The last class of extensions introduced in [30], namely the partial extensions, are defined for a default theory $\Delta$ and a well-ordering of $D$. The way the defaults of $D$ are applied is quite different from the way they are applied in our framework and lead to different conclusions.\(^\text{17}\) We note that Reiter extensions are partial extensions both under our semantics and this of [30] (given a suitable ordering of the defaults).

**Example 6.16 (partial extensions, Marek and Truszczynski [30]).** Consider the theory $\Delta = (D, W)$, with

$$W = \{A \lor B \lor C\} \quad \text{and} \quad D = \left\{ \frac{M \neg C}{A}, \frac{M \neg A}{B}, \frac{M \neg B}{C} \right\}.$$ 

This theory has one partial extension under our semantics, namely $E = \text{Th}(\{A \lor B \lor C\})$, and three partial extensions under the semantics of [30]. These extensions are

---

\(^{17}\) As far as the applicability of defaults is concerned, this semantics is closer to the approaches of [7, 38].
\(E_1 = \text{Th}(\{A\}), E_2 = \text{Th}(\{B\})\) and \(E_3 = \text{Th}(\{C\})\), and correspond to the different orderings of the rules.

In [1, 2] the extension class semantics was introduced. A particular extension class is defined to be the well-founded semantics of a default theory. Consider the following example.

**Example 6.17 (Extension classes, Baral and Subrahmanian [1, 2]).** Let

\[
W = \emptyset \quad \text{and} \quad D = \left\{ \frac{\text{MA} \cdot \text{M} \neg A \cdot \text{MB}}{A}, \frac{\neg A}{B}, \frac{\neg C}{C} \right\}.
\]

Then, \(E = \{\text{Th}(C)\}\) is a partial extension for this theory. However see that in the extension class \(E' = \{\text{Th}(C), \text{Th}(C, \neg A, A), \emptyset\}\), \(C\) cannot be assigned the value true because of the empty set. The other two partial extensions of \((D, W)\), namely \(E_1 = \text{Th}(\{A, C\})\) and \(E_1 = \text{Th}(\{\neg A, C\})\) are singleton extension classes for the theory.

The difference between extension classes and partial extensions is also evident in the case of well-founded extensions.

**Example 6.18 (Well-founded extensions, Baral and Subrahmanian [1]).** Consider the theory \(\Delta = (D, W)\), with

\[
W = \{A\} \quad \text{and} \quad D = \left\{ \frac{\text{A} \cdot \text{M} \neg K, \text{MB}}{A}, \frac{\text{A} \cdot \text{M} \neg C, \text{M} \neg B}{B}, \frac{\text{B} \cdot \text{MC, M} \neg D}{C \land \neg D} \right\}.
\]

Then the well-founded extension under the semantics of [1] is \(E = \text{Th}(\{A\})\). The only (maximal) partial extension of this theory, which coincides with the well-founded extension under our semantics, is \(E' = \text{Th}(\{A,B,C,\neg D\})\). Note that \(E\) is not a partial extension of \(\Delta\). However, \(E'\) is a singleton extension class of \(\Delta\).

Similarly, partial extensions are different from stationary extensions, introduced in [35]. Consider again the theory of Example 6.17.

**Example 6.19 (Stationary extensions, Przymusinska and Przymusinski [35]).** Let \(\Delta = (D, W)\) with

\[
W = \emptyset \quad \text{and} \quad D = \left\{ \frac{\text{M} \neg A \cdot \text{MA} \cdot \text{MB}}{\neg A}, \frac{\text{A}}{C} \right\}.
\]

This theory has three partial extensions including \(E = \{C\}\) which is the well-founded extension. However \(E\) is not a stationary extension of \(\Delta\). On the other hand, the empty set is the least stationary extension, but it is not a partial extension.

In contrast to the above mentioned approaches which are mainly proof-theoretic, in [34] a three-valued reconstruction of autoepistemic logic has been proposed. This semantics can be easily extended to default logic. We compare the semantic notion of
three-valued belief interpretations to the partial extensions by means of the following two examples.

Example 6.20 (Three-valued belief interpretations, Przymusinski [34]). Let $\Delta = (D, W)$ be a theory where

$$W = \emptyset \quad \text{and} \quad D = \left\{ \frac{M \neg A}{A}, \frac{M A}{\neg B} \right\}.$$ 

Theory $\Delta$ has only one partial extension $E = Th(\{ B \})$. Translate $\Delta$ into the autoepis- temic theory $T_\Delta = \{ A \leftarrow \neg L A, B \leftarrow \neg L \neg A \}$. Then, $E$, with $E(L A) = E(L \neg A) = E(L B) = E(L \neg B) = \text{undefined}$, is a three-valued belief interpretation. However the associated with $E$ set of propositions $E' = \emptyset$ is not a partial extension for $\Delta$.

Example 6.21 (Three-valued belief interpretations, Przymusinski [34]). Consider again the theory of Example 6.16,

$$W = \{ A \lor B \lor C \} \quad \text{and} \quad D = \left\{ \frac{M \neg C}{A}, \frac{M \neg A}{B}, \frac{M \neg B}{C} \right\}.$$ 

This theory has one partial extension under our semantics, namely $E = Th(\{ A \lor B \lor C \})$ but no three-valued belief interpretation.

7. Concluding remarks

In this paper we were concerned with extending the links between three fields of research, namely logic programming, default logic and graph theory.

Every normal logic program can be transformed into a graph. The stable, partial stable and well-founded semantics correspond to graph theoretic constructs, namely kernels, semikernels and the initial acyclic part, respectively. This graph representation offers several advantages. First, various results from pure and algorithmic graph theory can be employed in the investigation of both the theoretical and computational properties of logic programs. New classes of programs that always have stable models were obtained, and new algorithms that use the graph representation in the computations become possible. These methods compare favorably with other classical problem solving methods for logic programming (see [11,12] for details). Finally, the graph model gives a clear understanding of how interaction between rules can be resolved within different semantics.

We also presented a reconstruction of default logic based on a straightforward generalization of the semantics developed for logic programs. The problem of the non-existence of extensions was resolved in an intuitively appealing manner, whilst the deterministic and well-founded extensions provide a semantically strong background
for skeptical default reasoning. Not surprisingly, the graph structures defined for logic programs remain meaningful in the case of default theories.

Several directions for further research exist. Better algorithms for solving the kernel and semikernel problem are yet to be developed. Future results of the graph theoretic research on these problems, will probably be useful for the field of nonmonotonic reasoning. Furthermore the relation of the graph model with the theory of games can provide a useful link between the latter and logic programming. Finally, the default logic approach introduced in this paper can be applied to different domains that require default reasoning capabilities (e.g. inheritance networks).

Acknowledgements

The authors thank Pierre Duchet, Antonis Kakas, Oscar Meza, Hans Jürgen Ohlbach, Christos Papadimitriou and Jeffrey Ullman for their useful comments and discussions.

References
