MAX-PLANCK-INSTITUT
FÜR
INFORMATIK

Graph Theoretical Structures in Logic Programs and Default Theories

Yannis Dimopoulos Alberto Torres

MPI–I–93–264 November 1993

Im Stadtwald
66123 Saarbrücken
Germany

Graph Theoretical Structures in Logic Programs and Default Theories

Yannis Dimopoulos Alberto Torres

MPI–I–93–264 November 1993

Author's Address

Yannis Dimopoulos
Max-Planck-Institut für Informatik
Im Stadtwald,
66123 Saarbrücken, Germany,
e-mail: yannis@mpi-sb.mpg.de

Alberto Torres,
Stanford University,
Computer Science Department,
Stanford, CA 94305-2140, USA
e-mail: torres@cs.stanford.edu

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 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 the associated graph. This link allows us to consider both theoretical problems (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.

Keywords

Logic Programming, Semantics, Negation, Stable models, Partial Stable Models, Well-founded Semantics, Graph Theory, Complexity, Default Logic.

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 in capturing these pattern of reasoning in a formal representation. One of the most prominent nonmonotonic reasoning formalizations is *Default Logic*. One 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 with default logic. Thus logic programs with negation provide us a framework for nonmonotonic reasoning.

Some recent work has dealt with the relation between some nonmonotonic formalisms and graph-theoretic constructs. Torres shows in [Tor93a] that the stable models of logic programs correspond to the kernels of an associated graph. This result is extended in [Tor93b], which proves that the maximal semikernels of the same graph correspond to partial stable models. For disjunction-free default theories, Dimopoulos and Magirou show in [DM92] that extensions correspond to kernels in a related graph. In this paper, we further extend the forementioned results with the introduction of an 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 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 [GL88] correspond to kernels, partial stable models [Prz90, SZ90] to semikernels and the well-founded partial model [VRS88] to a special semikernel called the *initial acyclic part*.

We use the logic programming notion of *support*, which we extend to disjunction free default theories, to show that the above equivalences can be extended to the case of disjunction-free default theories. Aside from the theoretical interest of the above results, we believe the practical contribution of this paper is twofold. On the one hand, the 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 these interactions can be resolved. Furthermore they allow us to approach the formalizations in a way that ignores the logical meaning and concentrates on the structural properties of them. This is particularly useful when we try to investigate complexity issues or to tackle problems like the existence of particular semantics (stable models, extensions) and the development of algorithms.

On the other hand, the unified graph model give 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 as 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 semantic and graph-theoretic 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.

2.1 Logic Programs and Hypotheses

A *program* P is a set of first order rules of the form:

$$\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n \leftarrow \gamma_1 \wedge \gamma_2 \wedge \ldots \wedge \gamma_m$$

where $n \geq 1$, $m \geq 0$, every α_i is an atom, and and every γ_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:

$$\begin{gathered}\alpha_1 \leftarrow \gamma_1 \wedge \gamma_2 \wedge \ldots \wedge \gamma_m \\ \alpha_2 \leftarrow \gamma_1 \wedge \gamma_2 \wedge \ldots \wedge \gamma_m \\ \vdots \\ \alpha_n \leftarrow \gamma_1 \wedge \gamma_2 \wedge \ldots \wedge \gamma_m\end{gathered}$$

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.

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

Let P be a logic program. We denote by $\mathcal{H}(P)$ the Herbrand base of P and by $P^{\downarrow}$ 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 Λ is a set of literals then $\neg\Lambda$ is the set of literals corresponding to the negation of elements in Λ.

A hypotheses Δ *enables* a rule r if all negative subgoals in r are contained in Δ, that is, $(body(r) - \Delta) \subseteq \mathcal{H}(P)$. The set of rules n a program P enabled by a hypotheses Δ is denoted by $enabled(\Delta, P)$. We extend this notation to interpretations through the following definition $enabled(I, P) = enabled(I^-, P)$.

2.2 Supports and Attacks

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

Definition 2.1 *A hypothesis Δ is a* support *for an atom α in a program P (denoted by $\Delta \overset{P}{\mapsto} \alpha$)*[1] *if $P_\Delta^+ \models \alpha$. If $\Theta \subseteq \mathcal{H}(P)$, we write $\Delta \overset{P}{\mapsto} \Theta$ if for all $\alpha \in \Theta$ we have $\Delta \overset{P}{\mapsto} \alpha$. We denote by $\Delta^{\overset{P}{\mapsto}}$ the set of atoms supported by a hypothesis Δ. Furthermore, a support Δ is minimal for α (denoted by $\Delta \overset{\min,P}{\mapsto} \alpha$) if no subset of Δ supports α.*

Example 2.2 Consider the program P_1 following:

$$p \leftarrow \neg q \wedge \neg t$$
$$q \leftarrow \neg p \wedge \neg t$$
$$r \leftarrow \neg p$$
$$r \leftarrow \neg r$$
$$t \leftarrow s$$

$\{\neg q, \neg t, \neg p\}$ 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 Δ 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 $P_1 \cup \{\neg r\} \models p$.

[1] **Notation:** We omit the superscript P from the above notation as well as other introduced later when it is clear form the context. A min superscript over a binary relation always indicates the minimality of the left operand (with respect to set inclusion).

Definition 2.3 *A hypothesis* Δ attacks *another hypothesis* Δ' *in a program* P *(denoted by* $\Delta \overset{P}{\rightsquigarrow} \Delta'$*) if* $\Delta \overset{P}{\mapsto} \beta$ *for some* $\neg\beta \in \Delta'$*. A hypothesis* Δ *is* self-consistent *in a program* P *if it does not attack itself.*

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

Definition 2.4 *An assumption* $\neg\beta$ *is* unfounded *with respect to a hypothesis* Δ *if for every* Δ' *such that* $\Delta' \mapsto \beta$ *we have* $\Delta \rightsquigarrow \Delta'$*. We denote by* $\mathcal{U}_P(\Delta)$ *the set of all unfounded assumptions w.r.t.* Δ *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 α 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. We now define the semantical constructs explored in this paper:

Definition 2.5 *Let* P *be a program and* Δ *be a self-consistent hypothesis. The* supported interpretation *of* Δ *is* $I_\Delta = \Delta \cup \Delta^{\mapsto}$*. We say that an interpretation is* supported *if it is the supported interpretation of some self-consistent hypothesis* Δ*. 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 $\{\neg q, \neg t\} = I^- \subset \mathcal{U}_{P_1}(I^-) = \{\neg q, \neg s, \neg t\}$. 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: *if* I *is total.*

Partial stable model: *if* I *is a maximal well-founded interpretation.*

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

Well-founded (partial) model: *if* I *is the minimal deterministic model, which 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.[2]

Theorem 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 $I_{\mathcal{U}_P(I^-)}$ 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 $\Delta \mapsto \neg\beta$ then $\Delta \not\subseteq I^-$, because I^- is self-consistent. Since $\Delta \cup \neg I^+ \neq \emptyset$ and I is supported, $I^- \rightsquigarrow \Delta$. Therefore, $\neg\beta \in \mathcal{U}_P(I^-)$ and $I^- \subseteq \mathcal{U}_P(I^-)$. □

Theorem 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 $I^- \mapsto \beta$ 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 $\mathcal{U}(I^-)_P \subseteq I^-$. Since I is a well-founded interpretation, $I^- = \mathcal{U}_P(I^-)$. □

Since every stable model is total, the above theorem 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 Δ, we have $\Delta \overset{P_1}{\mapsto} \alpha$ if and only if $\Delta \overset{P_2}{\mapsto} \alpha$.*

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

2.4 Default Theories

In [Rei80] *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 $\Delta = (W, D)$, where W is a finite set of propositions and

[2] The proof of this latter fact was originally given in [SZ90].

D is a finite set of default rules of the form $d = a : Mb_1 \ldots Mb_n/w$, where $a, b_1, \ldots, b_n, w$ are arbitrary propositions. Proposition a is called prerequisite of the rule d (denoted as Prer(d)), the set of propositions $b_1, \ldots, b_n$ justifications (denoted as Just(d)) and the proposition w consequent (denoted as 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 an extension, which is intuitively what can consistently be believed given (W, D).

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

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.

2.5 Graphs and Kernels

A *directed graph* or *graph*[3] 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.

If $\mathcal{G}$ is a graph and $v \in \mathcal{V}(\mathcal{G})$ we define $\Gamma^+_{\mathcal{G}}(v) = \{v' | (v, v') \in \mathcal{E}(\mathcal{G})\}$, and $\Gamma^-_{\mathcal{G}}(v) = \{v' | (v', v) \in \mathcal{E}(\mathcal{G})\}$. These definitions are extended to sets of nodes through the following equations: $\Gamma^+_{\mathcal{G}}(V) = \bigcup_{v \in V} \Gamma^+_{\mathcal{G}}(v)$, and $\Gamma^-_{\mathcal{G}}(V) = \bigcup_{v \in V} \Gamma^-_{\mathcal{G}}(v)$. The subscript 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 $(\mathcal{V}', \mathcal{E}/\mathcal{V}')$.

The set of nodes of a graph which have no outgoing edges is called the set of *sink nodes* for the graph.

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 $\mathcal{E}/\mathcal{V}' = \emptyset$.*

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

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

[3] In this paper the term graph refers exclusively to directed graphs.

Definition 2.14 *Let $\mathcal{G}$ be a graph, and $\mathcal{K}$ a subset of $\mathcal{V}(\mathcal{G})$. We say that $\mathcal{K}$ is a* kernel *if it is independent and dominant. We also say that $\mathcal{K}$ is a* semikernel *if it is independent and semidominant.*[4]

The following proposition follows trivially from the above definition:

Proposition 2.15 *Let $\mathcal{G}$ be a graph. If $\mathcal{K}$ is a kernel of $\mathcal{G}$ then $\mathcal{K}$ is a maximal semikernel of $\mathcal{G}$.* □

Definition 2.16 *A graph $\mathcal{G}$ is* kernel-perfect *if for every $\mathcal{V}' \in \mathcal{V}(\mathcal{G})$ we the graph $\mathcal{G}/\mathcal{V}'$ 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:[5]

Proposition 2.17 *A graph 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 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be graph. We define the* initial acyclic segment *of $\mathcal{G}$ to be a set of independent nodes $\mathcal{V}' \subseteq \mathcal{V}$ such that it can be well-ordered in such a way that for every $v \in \mathcal{V}'$ we have $\Gamma^-(v) \subseteq \Gamma^+(\{v' \in \mathcal{V}' | v' < v\})$. The* initial acyclic part *of a graph is its maximal initial acyclic segment.* □

Lemma 2.19 *If $\mathcal{IS}$ is a set of initial acyclic segments of $\mathcal{G}$ then $\overline{\mathcal{IS}} = \bigcup_{S \in \mathcal{IS}} S$ is an initial acyclic segment of $\mathcal{G}$.*

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

[4] Often symmetric to our definitions are used for kernels and semikernels (see for example [Ber73]).

[5] For a more extensive review of the area see [BD90].

Theorem 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.

Theorem 2.21 *If IS is an initial acyclic segment of $\mathcal{G}$ then I is a semikernel of $\mathcal{G}$.*

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

$$\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n \leftarrow \neg\beta_1 \wedge \neg\beta_2 \wedge \ldots \wedge \neg\beta_m$$

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

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

Theorem 3.2 *Let P be a negative logic program. Then $\Delta \overset{P}{\mapsto} \alpha$ if and only if Δ enables a rule in P such that $\alpha \in \text{head}(r)$.*

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

Corollary 3.3 *Let P be a negative logic program. If $\Delta \overset{\min,P}{\mapsto} \alpha$ then there is a rule r in P such that $\Delta = body(r)$ 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. Δ if and only if for every rule r in P such that $\beta \in head(r)$ then $\Delta \overset{P}{\rightsquigarrow} body(r)$.* □

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 $body(r_1) = body(r_2)$ then $r_1 = r_2$.*

2. If $head(r_1) \cap head(r_2) \neq \emptyset$ *and* $body(r_1) \subseteq body(r_2)$ *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.

Theorem 3.6 *Let P be a reduced negative logic program. $\Delta \overset{\min,P}{\mapsto} \alpha$ if and only if there is a rule r in P such that $\Delta = body(r)$ 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 $\alpha \in head(r)$ and $\Delta = body(r)$. Then $\Delta \overset{P}{\mapsto} \alpha$. Now, if $\Delta' \overset{\min,P}{\mapsto} \alpha$ and $\Delta' \subseteq \Delta$, there is a rule r' such that $body(r') = \Delta' \subseteq \Delta = body(r)$. 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.[6] For any given negative logic program, a support-equivalent reduced negative logic program can in fact be obtained by "reducing" the given program. 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 $\exists r' \in P\ (body(r') = body(r))$ **then**
    remove r **from** P
    add $head(r)$ **to** $head(r')$
  end if
end for
(2) **for each** r **in** P
  for each α **in** $head(r)$
    if $\exists r' \in P\ (\alpha \in head(r') \wedge body(r') \subset body(r)$ **then**
      if $head(r) - \{\alpha\} \neq \emptyset$ **then**
        remove α **from** $head(r)$
      else
        remove r **from** P
      end if
    end if

[6] We generalize this result to general logic programs in Section 4.

end for

end for

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 $\mathcal{RG}(P)$) is the directed graph $(\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{r | r \in P\}$ and $\mathcal{E} = \{(r_1, r_2) | head(r_1) \cap \neg body(r_2) \neq \emptyset\}$.*[7]

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 $enabled(I, P)$ is a semikernel of $\mathcal{RG}(P)$.*

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

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

Proof: Let us first prove that $body(K)$ is not self defeating. If $body(K)$ is self-defeating then there is a rule r in $enabled(body(K), P)$ such that $\alpha \in head(r)$ and $\neg\alpha \in body(K)$. 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 $\mathcal{RG}(P)$. But this means that $body(r')$ is a minimal attack of an assumption $\neg\beta$ in $body(r) \subseteq body(K)$. Since $\neg\beta \in body(K)$, there is an $r'' \in K$ such that $\neg\beta \in body(r'')$. 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 $body(K) \subseteq \mathcal{U}_P(body(K))$. Let $\neg\beta \in body(K)$ and $\Delta \overset{\min,P}{\mapsto} \beta$. Then there is a rule r in P such that $body(r) = \Delta$. Therefore, there is an edge from r to some rule

[7] Notice that $\mathcal{E} = \{(r_1, r_2) | body(r_1) \overset{\min,P}{\rightsquigarrow} body(r_2)\}$.

in K, but since K is a semikernel there is an edge from some other rule r' in K to r. Then $body(r') \overset{\min,P}{\rightsquigarrow} \Delta$ and $body(K) \overset{P}{\rightsquigarrow} \Delta$. Therefore $\neg\beta$ is unfounded w.r.t. body(K). □

Theorem 3.10 is not the full reciprocal of Theorem 3.9 since there are many well-founded interpretations that are not of the form $I_{body(K)}$, where K is a kernel of $\mathcal{RG}(P)^{-1}$. 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.[8]

Definition 3.11 *Let P be a program and Δ a hypothesis. A subset Υ of $U_P(\Delta) - \Delta$ is an* addition set *for Δ in P if $(\Delta \cup \Upsilon)^{\overset{P}{\mapsto}} = \Delta^{\overset{P}{\mapsto}}$.*

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

1. $(I^-)^{\overset{P}{\mapsto}} = body(enabled(I,P))^{\overset{P}{\mapsto}}$
2. $U_P(I^-) = U_P(body(enabled(I,P)))$

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

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

Proof: Let us first prove the "if" part. Let K be a semikernel a of $\mathcal{RG}(P)$ and Υ an addition set for $body(K)$ in P. Using Theorem 3.10, we have that $I_{body(K)}$ is a well-founded interpretation. Now $I = I_{body(K)} \cup \Upsilon$ is supported since the assumptions in Υ do not support any new atom. And since $\Upsilon \subset U_P(body(K)) = U_P(body(K) \cup \Upsilon)$ then I is well-founded.
To prove the "only if" part, consider any well-founded interpretation I. By Theorem 3.9 we know that $enabled(I^-,P)$ is a semikernel of $\mathcal{RG}(P)$. Now, to prove that $\Upsilon = I^- - body(enabled(I,P))$ is an addition set for $body(K)$ in P we notice that since I is well-founded we have that $\Upsilon \subset U_P(I^-) - body(enabled(I,P))$. By Lemma 3.12 we have $U_P(I^-) = U_P(body(enabled(I,P)))$, so $\Upsilon \subset U_P(enabled(I,P)) - body(enabled(I,P))$. Lemma 3.12 also implies that $(I^-)^{\overset{P}{\mapsto}} = body(enabled(I,P))^{\overset{P}{\mapsto}}$. Therefore Υ is an addition set for $body(enabled(I,P))$. □

Corollary 3.14 *An interpretation I for a program P is a partial stable model of P if and only if there is a maximal semikernel K in $\mathcal{RG}(P)$ such that $I = I_{body(K)} \cup \Upsilon$ where $\Upsilon = U_P(body(K)) - body(K)$ is the maximal addition set for $body(K)$ in P.* □

Stable model impose stronger restrictions on the rules they enable. While rules enabled by a well-founded interpretation have to attack only rules which can attack them, the rules enabled

[8] In [Tor93a, Tor93b] a different approach is used. We discuss this approach in Section 4.

by the stable models have to attack any other rule. The next theorem formally states the link between stable models and kernels.

Theorem 3.15 *An interpretation I for a program P is a stable model of P if and only if there is a kernel K in $\mathcal{RG}(P)$ such that $I = I_{body(K)} \cup \Upsilon$ where $\Upsilon = \mathcal{U}_P(body(K)) - body(K)$ is the maximal addition set for $body(K)$ 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 $body(K)$) or all its rules are made invalid by $body(K)$ (hence it is unfounded w.r.t. $body(K)$). Therefore I is total.
To prove the "only if" part, consider $K = enabled(I, P)$. By the above corollary we know that K is a maximal semikernel. Since K is independent, if K is not a kernel then there is a rule $r \in P - K$ such that $r \notin \Gamma^+_{\mathcal{RG}(P)}(K)$. Since K is a maximal semikernel, $r \notin \Gamma^-_{\mathcal{RG}(P)}(K)$. But then the atoms in $head(r)$ are undefined in I, which contradicts the fact that I is a (total) stable model. □

The next two theorems demonstrate the fact that within well-founded models serious 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 their interactions, leading to a particular (and from a combinatorial point of view, rather simple) semikernel.

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 $I = I_{body(IP)} \cup \Upsilon$ where IP is the initial acyclic part of $\mathcal{RG}(P)^{-1}$ and $\Upsilon = \mathcal{U}_P(body(IP)) - body(IP)$ is the maximal addition set for $body(IP)$ in P.*

Proof: To prove that if $I = I_{body(IP)} \cup \Upsilon$ then I is the minimal complete well-founded interpretation of P, we first show that Υ is an addition set for $I_{body(IP)}$. Since IP is maximal then $\forall r \in P$, if $body(r) \subseteq \Upsilon \cup body(IP) = \mathcal{U}_P(body(IP)) \cup body(IP)$, then $r \in IP$ and body(r) $\subseteq$body(IP). Hence, $(\mathcal{U}_P(body(IP)) \cup body(IP))^{\overset{P}{\mapsto}} = body(IP)^{\overset{P}{\mapsto}}$ and Υ is an addition set for $I_{body(IP)}$. Since the initial acyclic part is a semikernel as well, from Theorem 3.13 follows that I is a well-founded interpretation. Additionally, since $\Upsilon \subseteq I$, I is complete. Moreover, every literal in Υ should be included in I for I to be complete. Additionally we can prove inductively on the well-order for IP that if literal $p \in body(IP)$ is omitted then I is not complete. Hence, I is the minimal complete well-founded interpretation of P, that is, its well-founded model.
Let I now be the the well-founded model of P. Since I is complete then $I_1 \subseteq I^-$, where $I_1 = \{\neg b | \nexists r \in P, b \in head(r)\}$. Let $R(I_1) = \{r | r \in P, body(r) \subseteq I_1\}$. Then $head(R(I_1)) \subseteq I$. Let $I_2 = \{\neg b | \forall r, b \in head(r), \exists \neg p \in body(r), p \in head(R(I_1))\}$. Then $I_2 \subseteq \mathcal{U}_P(I)$ and hence $I_2 \subseteq I^-$. Let $R(I_2) = \{r | r \in P, body(r) \subseteq I_1 \cup I_2\}$. Then again $head(R(I_2)) \subseteq I$. Iterating in this way over the ordinals we can prove that $I = I_{\bigcup_\alpha I_\alpha} = I_{\bigcup_\alpha body(R_\alpha)} \cup (\mathcal{U}_P(I_{\bigcup_\alpha body(R_\alpha)}) - \bigcup_\alpha body(R_\alpha))$.

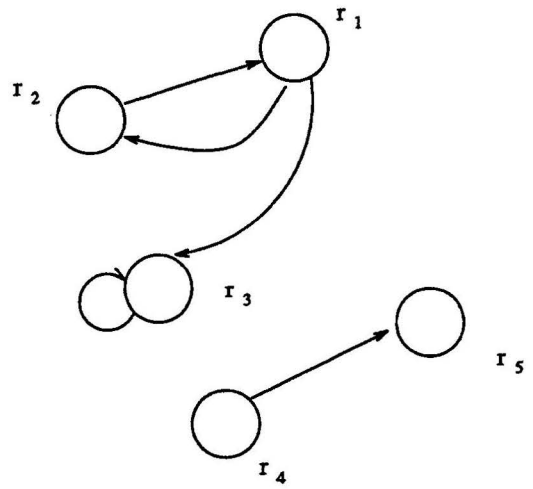


Figure 1: Rule graph for P_2 ($\mathcal{RG}(P_2)$)

But since the set $\bigcup_\alpha R_\alpha$ complies with definition 2.18, it is the initial acyclic part IP for $\mathcal{RG}(P)$ and therefore $I = I_{body(IP)} \cup \Upsilon$. □

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

$$p, s \leftarrow \neg q, \neg r$$

$$q \leftarrow \neg p, \neg r$$

$$s \leftarrow \neg s$$

$$t \leftarrow \neg r$$

$$u \leftarrow \neg t$$

The rule graph of P_2, $\mathcal{RG}(P_2)$, is depicted in the Figure 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 $I_1 = \{\neg q, \neg r, p, s, t\}$, $I_2 = \{\neg p, \neg r, q, t\}$, $I_3 = \{\neg r, t\}$, respectively. None of these well-founded interpretations is complete. The corresponding complete well-founded models are $I_4 = \{\neg q, \neg r, p, s, t, \neg u\}$, $I_5 = \{\neg p, \neg r, q, t, \neg u\}$, $I_6 = \{\neg r, t, \neg u\}$. 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 $\mathcal{RG}(P)$ then I_4 is a stable model as well. Finally the initial acyclic part of $\mathcal{RG}(P)$ is the set S_3, hence the well-founded model of P is the set $I_6 = \{\neg r, t, \neg u\}$. □

4 The Case of General Logic Programs

In this section, we extend the results introduced for negative logic programs to 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* body(r) *is a minimal support of some atom in P, and* head$(r) = \{\alpha | \Delta \overset{\min,P}{\mapsto} \alpha\}$.

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

Proof: Since no two different rules have the same body in the transformed program fulfills Condition *1* of Definition 3.5. Now, if $\alpha \in head(r_1) \cap head(r_2)$, it is not possible that $body(r_1) \subset body(r_2)$ because otherwise $body(r_2)$ would not be a minimal support of α. Therefore P^- is reduced.
The fact that P^- is support-equivalent to P follows directly from Theorem 3.6. □

Theorem 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. □

We now introduce the minimal support 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 $\mathcal{MAG}(P)$, is the directed graph $(\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{\Delta | \exists \alpha \, \Delta \overset{\min,P}{\mapsto} \alpha\}$ and $\mathcal{E} = \{(\Delta_1, \Delta_2) | \Delta_1 \overset{\min,P}{\rightsquigarrow} \Delta_2\}$.*

Theorem 4.5 *Let P be a logic program. The graph $\mathcal{MAG}(P)$ is isomorphic to the graph $\mathcal{RG}(P^-)$.*

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. □

Combining the above theorem 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 $\mathcal{MAG}(P)^{-1}$ such that $I = I_{body(K)} \cup \Upsilon$ where Υ is an addition set for $body(K)$ in P.*

2. *I is a partial stable model of P if and only if there is a maximal semikernel K in $\mathcal{MAG}(P)^{-1}$ such that $I = I_{body(K)} \cup \Upsilon$ where $\Upsilon = U_P(body(K)) - body(K)$ is an addition set for $body(K)$ in P.*

3. *I is a stable model of P if and only if there is a maximal kernel K in $\mathcal{MAG}(P)^{-1}$ such that $I = I_{body(K)} \cup \Upsilon$ where $\Upsilon = U_P(body(K)) - body(K)$ is an addition set for $body(K)$ in P.*

4. *I is the well-founded model of P if and only if $I = I_{body(IP)} \cup \Upsilon$ where IP is the initial acyclic part of $\mathcal{MAG}(P)^{-1}$ and $\Upsilon = U_P(body(IP)) - body(IP)$ is an addition set for $body(IP)$ 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 $\mathcal{MAG}(P)$ 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 α in P, there is a proof of α that uses a rule with $\neg\beta$ in its body. It follows that α 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:*

$$p \leftarrow q \wedge r$$

$$q \leftarrow \neg p$$

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

Definition 4.11 *A program P is* odd-loop free *if every cycle in $\mathcal{MAG}(P)$ is of even length.*

Proposition 4.12 *Every odd-loop free program has at least one stable model.* □

Theorem 4.13 *Every structurally total program is odd-loop free.*

Proof: Since there are no odd-length loops at the predicate level. If follows trivially that there can no be any odd-length loop at the atom (support) level. □

Proposition 4.14 *Let P be a logic program. If $\mathcal{MAG}(P)$ is symmetric then P has at least one stable model.* □

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 has to address. This section is devoted in demonstrating how graph theory can contribute in obtaining complexity results, determining cases where reasoning is tractable, and defining new notions of approximation.

To begin with, given the intractability of determining whether a graph has a kernel ([GJ79]), the problem of determining whether a negative logic program possesses stable models is NP-complete.[9] On the other hand semikernels, or equivalently well-founded interpretations and partial stable models, are more easy to be found in a graph. 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 sightly different decision problems.

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

The next theorem states that this problem is intractable.

Theorem 5.1 *The above decision problem is NP-complete.*

Proof: The proof is by reduction from 3-SAT. Given a formula in CNF $C = \{C_1, C_2, \ldots, C_n\}$, $C_i = C_{i1} \vee C_{i2} \vee C_{i3}$ we construct a graph $G = (N, E)$, like the one shown in Figure 1 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 with the name 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) A bidirectional edge, between the nodes which correspond to complementary literals.

2) For every literal x_i occurring in clause C_j an edge (n_i, c_j).

3) From every $c_i \in S$ an edge (c_i, Aux).

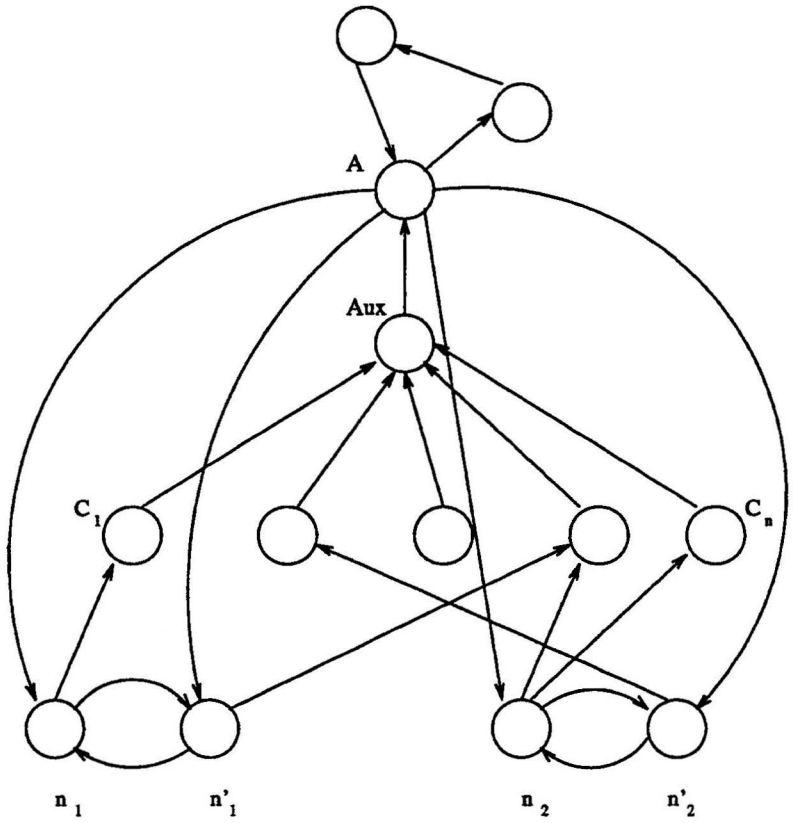
4) The edge (Aux, A), as well as an edge (A, n_i) from A to every node $n_i \in L$.

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

Proof: Let SK be a semikernel of the graph, $SK <> \{\}$. Assume that SK contains a 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 nodes in L belong to SK and cover them. But then these nodes receive an edge form A which can not be dominated. Also A itself can not cover these nodes since it can not belong to SK. Hence no semikernel can contain a node from S. Hence every nontrivial semikernel must contain Aux and must not contain A.

[9] The complexity results in [MT91] regarding autoepistemic logic also imply this result.

What a semikernel must contain is a subset of the nodes of L which must cover every node in S. It is easy to see that such a semikernel is kernel.
Hence in G every nontrivial semikernel is a kernel. □



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. □

Lemma 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.* □

The next question we address is under which conditions an algorithm which computes kernels, is useful in computing semikernels as well.

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

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

Proof: Let K be a kernel for G'. Then the nodes in $K - N'$ are independent and for every node n_j such that $n_j \in \Gamma_G^-(n_i)$, $n_i \in K - N'$ there is 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 is 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 which computes kernels is also capable of computing semikernels if it is supplied with the semikernel equivalent of the graph at hand.

Clearly in the case of graphs without cycles the computation of the unique kernel, which coincides with the maximal semikernel, is trivial. This 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-loop free, and a kernel always exists. In this case we can perform the tie-breaking procedure introduced in [PY92] and compute nondeterministically in polynomial time a kernel of the graph. Furthermore as it shown in [DMP93] there is a polynomial delay algorithm[10] which enumerates a set of kernels for this class of graphs. Unfortunately this procedure is not complete, that is, there are kernels that will not be detected by the procedure. Nevertheless this procedure may serve as a sound but incomplete procedure for logic programs without odd cycles. It is also shown in [DMP93] that determining whether there is another kernel for such a graph, except from those found by this procedure is NP-complete. Finally skeptical reasoning with these graphs (i.e. the problem whether there is a node contained in all kernels) was also proved to be intractable in [DM92].

In the case of graphs with odd cycles there are still some possibilities 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 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 the kernels in the odd-cycle-free graph. The procedure is sound but incomplete.

We also note that if the graph is symmetric, then the set K is a kernel iff K is a *maximal independent set* 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 which computes *all* the MIS of a graph (see [TIAS77], [JPY88]).

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

[10] We say that an algorithm for generating configurations is *polynomial delay* [JPY88] if there is only a polynomial delay between any two configurations generated. Such algorithms may behave exponentially because of the number of the exponentially many different configurations, but this is obviously unavoidable.

easy to find a measure of the approximation. Some recent attempts include the approximate entailment of [CS92b] referring to default logic and circumscription, which extents their previous work in [CS92a] regarding classical logic. However 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. Under this view, given a graph G, the set S_1 is a better approximation than S_2, if $S_2 \subseteq S_1$, where $S_1, S_2 \subseteq S$ and S is a maximal semikernel for G.

6 The Case of Default Theories

Since Reiter's original definitions of default theories and their extensions, several researchers have given different definitions especially of the notion of the extension,[11] as well as of the notion of the default rule itself (e.g. [GLPT91]). 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 e.t.c.). On the other hand a number of researchers has applied the various semantics for logic programs, to default logic and other nonmonotonic formalisms. An early attempt towards this direction was described in [Prz89] where the well-founded semantics were defined for default and autoepistemic theories, based on a three-valued reconstruction of those formalisms. More recently well-founded semantics for the same formalisms were proposed in [BS91] and [BS92] based on an ordering for the sets of interpretations around which Gelfond-Lifschitz operator oscillates. In [PAA92] another reformulation of default logic is presented that satisfies some criteria defined by the authors, and which is along the lines of stable models for extended logic programs. In [PP92] the stationary extensions are presented, an extension of the stationary semantics for logic programs. Finally the approach of [Kak92] is closely related to the framework developed in this section.

In this section we extent the notions defined in section 2 for logic programs to the case of *propositional* default theories. This leads to several definitions of the notion of extension, which capture different methods (or modes) of reasoning with default logic. This generalization also links the graph theoretic results presented in the previous sections, 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 $A : MB \wedge C/C$, where $B \wedge C$ 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 some of the notions to the general case of propositional default theories. Notice that in the case of

[11] See [FJ92] for a general framework where several invariants of default logic are examined.

conjunctive seminormal theories, $MB \wedge C$ is equivalent to $MB \wedge MC$. Hence we can also use the term seminormal, for theories of conjunctive defaults of the form $a : Mb_1 \wedge \ldots \wedge Mb_n/w$, where b_i is a consistent conjunction and $w \subseteq b_1 \cup \ldots \cup b_n$.

Let $\Delta = (D, W)$ be a conjunctive default theory and let $D = \{d_1, \ldots d_n\}$. Any literal $p \in$Just$(d_1) \cup \ldots$Just(d_n) 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 Δ_H^+, where H is a hypothesis, the propositional theory $W \cup D'$ where D' is obtained by deleting from the set Just(d_i) of every rule d_i, the justifications in H, and then deleting every rule d_j, Just$(d_j) \neq \emptyset$. Notice that Δ_H^+ can be inconsistent.

A hypothesis H *supports* a literal α in Δ (denoted by $H \overset{\Delta}{\mapsto} \alpha$) if $\Delta_H^+ \models \alpha$ and in Δ_H^+ not both a literal and its negation occur in the heads of the rules in D' or in W. A hypothesis H *attacks* another hypothesis H' in a theory Δ (denoted by $H \overset{\Delta}{\rightsquigarrow} H'$) if $H \overset{\Delta}{\mapsto} \beta$ for some $\neg\beta \in H'$.

An assumption β is *unfounded* with respect to a hypothesis H if for every H' such that $H' \mapsto \beta$ we have $H \rightsquigarrow H'$. We denote by $\mathcal{U}_\Delta(H)$ the set of all unfounded assumptions w.r.t. H in a theory Δ. We say that a set of propositions P is *supported* if there is a hypothesis H that supports every proposition in P. If for such a set P and its associated hypothesis H, $\neg H = \mathcal{U}_\Delta(H)$ holds as well, then P is a *partial* extension. The next theorem provides an alternative definition for the partial extensions in the vein of [Rei80].

Theorem 6.1 *A set of propositions E is a partial extension for a propositional seminormal conjunctive default theory $\Delta = (D, W)$ iff $E = \cup_{i=0}^{\infty} E_i$ for a sequence of sets E_i such that*
$E_0 = W$ and
$\forall i, i > 0$ $E_i = Th(E_{i-1}) \cup \{w | a : Mb/w, a \in E_{i-1}$, b is consistent with E and for every b_i, in $b = \bigwedge b_i$, $\neg b_i \in B_E\}$,
where B_E[12] $= \{p |$ if $p \notin W$ and for every sequence of defaults $d_1, \ldots, d_n \in D$, such that $p \in Cons(d_n)$, and for every d_i in the sequence, $Prer(d_i) \subseteq \cup_{j=1}^{i-1} Cons(d_j) \cup W$, the condition $\exists q \in Just(d_i)$ such that $\neg q \in E$ 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 $\mathcal{U}_\Delta(H) = B_E$. If $b \in \mathcal{U}_\Delta(H)$ then either there is no sequence of defaults concluding b, or for every such sequence there is a proposition $p_i \in$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 $b \notin \mathcal{U}_\Delta(H)$ then $b \notin B_E$. Hence $\mathcal{U}_\Delta(H) = B_E$. For every $q \in E$ there is a sequence of defaults $d_1, \ldots, d_k$ such that Prer$(d_i) \subseteq \cup_{j=1}^{i}$Cons$(d_j)$, for $1 \leq i \leq k$, and for every $p \in$Just(d_i) for every d_i in the sequence $p \in H$. Since $\mathcal{U}_\Delta(H) = B_E$ and $\mathcal{U}_\Delta(H) = \neg H$ then $\neg p \in B_E$. Hence every $q \in E$ satisfies the conditions of the theorem. On

[12] The full notation is $B_{E,\Delta}$ but we omit Δ form the subscript when we refer to exactly one theory

the other hand if $q \notin E$ then q is not supported by H. This means that for every sequence of rules providing q 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 have to prove that $U_\Delta(\neg B_E) = B_E$. Assume that $b \in U_\Delta(\neg B_E)$. Then either there is no sequence of defaults that concludes 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 concluding b (if one exists) is blocked by some $c \in E$, in other words some literals in $\neg B_E$. Hence $b \in U_\Delta(\neg B_E)$, and $U_\Delta(\neg B_E) = B_E$, which means that 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 justification with the partial extension, is necessary in order to avoid inconsistent extensions. Consider, for example, the theory $W = \emptyset$ and $D = \{: MB/B, M\neg B/\neg B\}$. 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. The next example demonstrates the difference between partial extensions and Reiter's extensions.

Example 6.2 *Let $\Delta = (D, W)$ be a default theory, where $W = \{A\}$ and $D = \{A : MB/B, A : M\neg B/\neg B, A : MC \wedge \neg B/C, : ME \wedge \neg G \wedge \neg C/E, : MF \wedge \neg E/F, : MG \wedge \neg F/G\}$. Theory Δ has two maximal partial extension, namely $E_1 = \{A, \neg B, C, F\}$ and $E_2 = \{A, B\}$. Notice that E_1 a Reiter-type extension as well, while E_2 is not, due to the presence of the last three rules.*
□

For every conjunctive default theory there is a partial extensions preserving transformation, to a conjunctive prerequisite-free default theory.

Theorem 6.3 *Let Δ be a conjunctive default theory and let Δ^- be the prerequisite free conjunctive default theory containing exactly the rules r' with $Cons(r') = \{\alpha\}$ and $Just(r')$ a minimal support for α, for every literal $\alpha \in Cons(r)$, $r \in \Delta$. Then E is a partial extension for Δ iff E is a partial extension for Δ^-.*

Proof (sketch): Let E be a partial extension of $\Delta = (D, W)$. We show that E is an extension for $\Delta^- = (D^-, W)$ as well, and furthermore $B_{E,\Delta} = B_{E,\Delta^-}$. We first show the second equality. Assume that $p \in B_{E,\Delta}$. There are two case 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, $\text{Prer}(d_i) \subseteq \cup_{j=1}^{i-1} \text{Cons}(d_j) \cup W$. See that in this case there is no support for P and there will be no rule in D^- with head p hence $p \in B_{E,\Delta^-}$. The second possibility is that for such sequence of default in Δ, there exists a $q \in E, \neg q \in Just(d_i)$ for some rule d_i in the sequence. But then

every rule in Δ^- concluding p will also contain such a proposition, hence $p \in B_{E,\Delta^-}$. Similar arguments can be constructed for the opposite direction, hence $B_{E,\Delta} = B_{E,\Delta^-}$.
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 $p \in$Just$(d_1) \cup \ldots \cup$Just(d_n) then $\neg p \in B_{E,\Delta}$. But then there will be a rule $d_k \in D^-$, Just$(d_k) =$Just$(d_1) \cup \ldots \cup$Just(d_n), with $b \in$Cons(d_k). Since $B_{E,\Delta} = B_{E,\Delta^-}$, $b \in E_{\Delta^-}$ as well. Also see that if a proposition $b \notin E_\Delta$, then for every rule $d_i \in D$, such that $b \in$Cons(d_i) either $\exists p \in$Prer$(d_i), p \notin E$ or $\exists p \in$Just$(d_i), \neg p \notin B_{E,\Delta}$. In both cases this means that $b \notin E_{\Delta^-}$ as well. Hence if E is an extension of $\Delta = (D, W)$ then E is an extension of $\Delta^- = (D^-, W)$.
We now show the opposite direction. Suppose that E is an extension for a theory Δ^-. First, using similar to the above given arguments, we can prove again that $B_{E,\Delta} = B_{E,\Delta^-}$. See that $b \in E_{\Delta^-}$, in terms of the theory Δ means that there must be a sequence of defaults in $d_1, \ldots, d_k \in D$ such that $b \in$Cons(d_k) and for every $p \in$Just$(d_1) \cup \ldots \cup$Just(d_n), $\neg p \in B_{E,\Delta}$. Furthermore see that if $q \in$Prer(d_i) for some d_i in the sequence, then $q \in$Cons(d_n). 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,Δ^-}. Then if $b \notin E_{\Delta^-}$ then $b \notin E_\Delta$ as well. □

It is important to stretch the fact 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 $W = \{A\}$ and $D = \{A : MB/B, A : M\neg B \wedge C/C, B \wedge C : MD/D\}$. The associated prerequisite free theory is $W = \{A\}$ and $D' = \{: MB/B, : M\neg B \wedge C/C, : MB \wedge M\neg B \wedge MC \wedge D/D\}$.

In [DM92], Reiter's extension of a prerequisite-free conjunctive default theory were proved to correspond to the kernels of the theory's graph. The next two theorems state the fact that partial extensions correspond to the semikernels of the same graph.

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

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

Proof (sketch): Let E be a partial extension for the prerequisite-free conjunctive default theory Δ. We will show that for $S = \{r : Just(r) \subseteq \neg B_E\}$, $S = S_1 \cup S_2$ 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 $\mathcal{RG}(\Delta)$, and the initial acyclic part of $\mathcal{RG}(\Delta) - (S \cup \Gamma^+(S))$

is empty. If a node $r_i \in S$ is receiving an incoming edge from some other node r_j in $\mathcal{RG}(\Delta)$, this means that there is a literal in 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 $\Gamma^-(S)$. On the other hand since for every other rule $m \notin S$ there is always a proposition $p_i \in$Just(m), $p_i \notin \neg B_E$, then the node m will receive some edges form some nodes not dominated by S, hence the initial acyclic part of $\mathcal{RG}(\Delta) - (S \cup \Gamma^+(S))$ will be empty. □

Theorem 6.6 *Let Δ be a prerequisite-free conjunctive default theory and $\mathcal{RG}(\Delta)$ its rule graph. Then there is partial extension E for Δ, for which $\neg B_E = S_1 \cup S_2 \cup S_3$, where*

1. $S_1 = \{b : b \in Just(r)$, *where either* $r \in K, K$ *is a semikernel of* $\mathcal{RG}(P)$ *or* $r \in IP$ *where* IP *is the initial acyclic part of* $\mathcal{RG}(\Delta) - (K \cup \Gamma^+(K))\}$
2. $S_2 = \{\neg b : b$ *occurs nowhere in the consequents of the rules of* $\Delta\}$.
3. $S_3 = \{\neg b : \forall r, b \in Cons(r), r \in \Gamma^+(K) \cup \Gamma^+(IP)\}$

Proof (sketch): The partial extension E will be the set supported by S. Hence we have to prove that $\neg S = \mathcal{U}(S)$. First see that every proposition b for which $b \notin$Cons(r), b is unfounded and hence if $\neg b \in S_2$ then $b \in \mathcal{U}(S)$. Let $b \in S_1$, and $b \in$Just(r_i) for some rule $r_i \in K$. Then if $\neg b \notin$Cons(r) for some rule $r \in \Delta$, $\neg b \in \mathcal{U}(S)$. Let $r_j \in D$ be a rule with $\neg b \in$Cons(r_j). The $(r_j, r_i) \in \mathcal{E}$. Since K is a semikernel there must be rule $r_k \in K$, $(r_k, r_j) \in \mathcal{E}$, Just($r_k$) $\in S_1$. This means that $\neg b \in \mathcal{U}(S_1)$, hence $\neg b \in \mathcal{U}(S)$. Let now $b \in S_1$, and $b \in$Just(r_i) for some rule $r_i \in IP$. For set IP there is well-order $IP_1, IP_2, \ldots$. Then the nodes in IP_1 are not receiving any edge except form nodes dominated by K, and every IP_i, for $2 \leq i$ is not receiving any edge except from nodes in $\Gamma^+(IP_{i-1})$. Then $\neg b \in \mathcal{U}(S_1)$, hence $\neg b \in \mathcal{U}(S)$. Finally for every proposition $b \in S_3$, $\neg b \in \mathcal{U}(S)$ holds. □

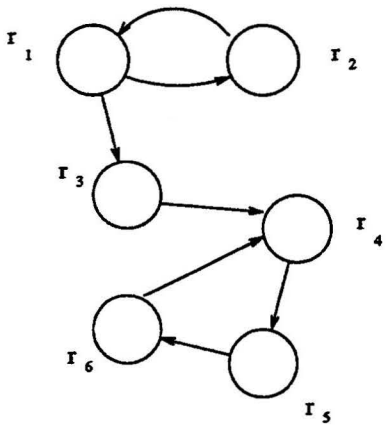


Figure 3

Example 6.7 *Consider the theory Δ of the example 6.2. We first convert the theory to one without prerequisites, obtaining in this way the theory $\Delta' = (D', W)$, where $D' = \{: MB/B, : M\neg B/\neg B, : MC \wedge \neg B/C, : ME \wedge \neg G \wedge \neg C/E, : MF \wedge \neg E/F, : MG \wedge \neg F/G\}$. The rule graph of this theory is depicted in figure 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 a (Reiter) extension as well. □

Theorem 6.1 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 $E = \cup_{i=0}^{\infty} E_i$ for a sequence of sets E_i such that*
$E_0 = W$ and
$\forall i, i > 0$ $E_i = Th(E_{i-1}) \cup \{w | a : Mb_1 \ldots Mb_n / w, a \in E_{i-1}$ and for every $b_i, 1 \leq i \leq n$, $b_i = \bigwedge b_{ij}$ where b_{ij} is disjunction of literals, b_i is consistent with E and $\neg b_{ij} \in B_E\}$,
where $B_E = \{p |$ if for every sequence of defaults $d_1, \ldots, d_n \in D$, $n \geq 0$, such that $Th(W \bigcup \cup_{j=1}^{n} Cons(d_j)) \vdash p$, and for every default d_i in the sequence $Pre(d_i) \subseteq Th(W \bigcup \cup_{j=1}^{i-1} Pre(d_j))$, the condition $\exists q \in Just(d_i)$ such that $\neg q \in E$ holds $\}$. □

Even though partial extensions are a reasonable semantics for default theories, there are cases where we would prefer to be more skeptical. We can capture this skepticism in default reasoning by means of *deterministic* and *well-founded* extensions, which are notions developed earlier for logic programs.

Definition 6.9 *Let Δ 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 Reiter-type characterization, for general propositional theories.

Theorem 6.10 *A set E is a well-founded extension for a default theory $\Delta = (D, W)$ iff $E = \cup_{i=0}^{\infty} E_i$ for a sequence of sets E_i such that*
$E_0 = W$ and
$\forall i, i > 0$ $E_i = Th(E_{i-1}) \cup \{w | a : Mb_1 \ldots Mb_n / w, a \in E_{i-1}$ 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 $\neg b_{ij} \in B_{E_i}\}$,
where $B_{E_i} = \{p |$ if for every sequence of defaults $d_1, \ldots, d_n \in D$ such that $Th(W \bigcup \cup_{j=1}^{n} Cons(d_j)) \vdash$

p, and for every default d_i in the sequence $Prer(d_i) \subseteq Th(W \bigcup \cup_{j=1}^{i-1} Prer(d_j))$, the condition $\exists q \in Just(d_i)$ such that $\neg q \in Th(E_{i-1})$ 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. □

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 see that the consistency check of the justification with the well founded extension is redundant. Consider for example the theory $W = \{A\}$ and $D = \{A : MD/B, A : MF/\neg B\}$. If we omit the consistency check of the justifications against 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 $\Delta = (D, W)$, where $W = \{A\}$ and $D = \{A : M\neg K \wedge B/B, A : M\neg C \wedge \neg B/\neg C, B : MC \wedge \neg D/C \wedge \neg D, C : MD \wedge \neg F/\neg F, C : MF \wedge E \wedge G/E \wedge G, C : M\neg E \wedge \neg G/\neg E \wedge \neg G, C : ME \wedge \neg G \wedge H/H, C : M\neg H/\neg H\}$.*
Theory Δ has two maximal partial extensions -which are Reiter's extensions as well- 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 Δ is the set DE_1. □

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 to a graph and its stable, partial stable and well-founded semantics correspond to graph-theoretic constructs, namely kernels, semikernels and the initial acyclic part. This graph representation gives a clear understanding of how interaction between rules can be resolved within different semantics. Furthermore we employed various results from pure and algorithmic graph theory and obtained in this way theoretical and computational interesting subclasses of programs.

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. On the other hand 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 as well. Similar considerations are possible for the autoepistemic logic

and various forms of truth maintenance systems (TMS). This offers us the possibility to answer questions regarding all these formalisms in a unified manner.

Acknowledgements The authors want to thank Pierre Duchet, Oscar Meza, Hans Jürgen Ohlbach, Christos Papadimitriou and Jeffrey Ullman for their useful comments.

References

[BD90] C. Berge and P. Duchet. Recent problems an results about kernels in directed graphs. *Discrete Matematics*, 86:27–31, 1990.

[Ber73] C. Berge. *Graphs and Hypergraphs.* North Holland, 1973.

[BS91] C. Baral and V.S. Subrahmanian. Dualities between alternative semantics for logic programs and nonmonotonic reasoning (extended abstact). In A. Nerode, W. Marek, and V. S. Subrahmanian, editors, *Logic Programming and Nonmonotonic Reasoning: Proc. of the 5th International Workshop*, pages 69–86, Washington, D.C., 1991. MIT Press.

[BS92] C. Baral and V. S. Subrahmanian. Stable and extensions class theory for logic programs and default logics. *Journal of Automated Reasoning*, 8:345–366, 1992.

[CS92a] M. Cadoli and M. Schaerf. Approximate entailment. In *Trends in AI: Proc. of the 2^{nd} Conference of the Italian Association for Artificial Intelligence*, pages 68–77, 1992. Springer Verlag LNAI 549.

[CS92b] M. Cadoli and M. Schaerf. Approximate inference in default logic and circumscription. In *Fourth International Workshop on Nonmonotonic Reasoning*, Plymouth, VT, 1992.

[DM92] Y. Dimopoulos and V. Magirou. A graph-theoretic approach to default logic. To appear in *Information and Computation*, 1992.

[DMP93] Y. Dimopoulos, V. Magirou, and C. Papadimitriou. On kernels, defaults and even graphs. Technical Report MPI-I-93-226, Max-Planck-Institut für Informatik, 1993.

[FJ92] C. Froidevaux and Mengin J. A framework for default logics. In D. Pearce and G. Wagner, editors, *Proc. European Workshop JELIA '92*, pages 154–173, Berlin, 1992. Springer Verlag, LNCS 633.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness.* W. H. Freeman and Company, New York, 1979.

[GL88] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *Proc. Fifth International Conference and Symposium on Logic Programming*, pages 1070–1080, Cambridge, Mass., 1988. MIT Press.

[GLPT91] M. Gelfond, V. Lifschitz, H. Przymusinska, and M. Truszczynski. Disjunctive defaults. In J. Allen, R. Fikes, and E. Sandewall, editors, *Proc. of the Second Int. Conf. on Principles of Knowledge Representation and Reasoning*, pages 230–237. Morgan Kaufman, 1991.

[JPY88] D. Johnson, C. Papadimitriou, and M. Yannakakis. On generating all maximal independent sets. *Information Processing Letters*, 27(3):119–123, 1988.

[Kak92] A. Kakas. Default reasoning via negation as failure. In G. Lakemeyer and B. Nebel, editors, *ECAI-92 Workshop on "The Theoretical Foundations of Knowledge Representation and Reasoning"*, 1992.

[MT91] A. Marek and M. Truszczynski. Autoepistemic logic. *Journal of the ACM*, 38(3):588–619, 1991.

[PAA92] L. M. Pereira, J. J. Alferes, and J. N. Aparicio. Default theory for well founded semantics with explicit negation. In D. Pearce and G. Wagner, editors, *Logics in AI: Proc. of the European Workshop JELIA'92*, pages 339–356. Springer, Berlin, Heidelberg, 1992.

[PP92] H. Przymusinska and T. Przymusinski. Stationary default extensions. In *Fourth International Workshop on Nonmonotonic Reasoning*, Plymouth, VT, 1992.

[Prz89] T. Przymusinski. Three-valued non-monotonic formalisms and logic programming. In R. Brachman, H. Levesque, and R. Reiter, editors, *Proc. First International Conference on Principle of Knowledge Representation and Reasoning*, pages 341–348, Toronto, Ontario, 1989.

[Prz90] T. Przymusinski. Extended stable semantics for noraml and disjunctive programs. In *Proc. of the Seventh International Conference on Logic Programming*, pages 459–477. MIT Press, 1990.

[PY92] C. Papadimitriou and M. Yannakakis. Tie-breaking semantics and structural totality. In *Proceedings Eleventh Symposium on Principles of Database Systems*, pages 16–22, 1992.

[Rei80] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[SZ90] D. Saccà and C. Zaniolo. Stable models and non-determinism in logic programs with negation. In *Proceedings Ninth Symposium on Principles of Database Systems*, pages 205–217, 1990.

[TIAS77] S. Tsukiyama, M. Ide, H. Ariyoshi, and I. Shirakawa. A new algorithm for generating all maximal independent sets. *SIAM J. Comptut.*, 6(3):505–517, 1977.

[Tor93a] A. Torres. Negation as failure to support. In *Proceedings of the Second Workshop on Logic Programming and Non-monotonic Reasoning*, 1993. (to appear).

[Tor93b] A. Torres. A nondeterministic well-founded semantics. Submitted to PDK'93, 1993.

[VRS88] A. Van Gelder, K. A. Ross, and J. S. Schlipf. Unfounded sets and well-founded semantics for general logic programs. In *Proceedings Seventh Symposium on Principles of Database Systems*, pages 221–230, 1988.