NORTH-HOLLAND

AN ARGUMENTATION-THEORETIC FOUNDATION FOR LOGIC PROGRAMMING*

PHAN MINH DUNG

▷ Logic programs are considered as abductive programs with negative literals as abductive hypotheses. A simple framework for semantics of logic programming is introduced based on the notion of acceptable hypotheses. We show that our framework captures, generalizes, and unifies different semantic concepts (e.g., well-founded models, stable models, stationary semantics, etc.) in logic programming. We demonstrate that our framework accommodates in a natural way both the minimalism and maximalism intuitions to semantics of logic programming. Further, we show that Eshghi and Kowalski's procedure is a proof procedure for the abductive semantics. We also give sufficient conditions for the coincidence between different semantics. ◁

INTRODUCTION

For a successful application of logic programming as a paradigm for knowledge representation, it is necessary to clarify the semantic problems of negation in logic programming and its relations to nonmonotonic logic. This paper presents a contribution to the study of this problem. Our goal is to reveal the inherent relations between abduction and logic programming.

To a first approximation, the semantics of a logic program may be defined by its Clark's completion [5, 28]. Given a logic program P, the completion of P, comp(P), consists of some equality axioms plus a completed definition of each

*A shorthand version of this paper has been published in [9].

Address Correspondence to: Computer Science Program, School of Advanced Technology, Asian Institute of Technology, P. O. Box 2754, Bangkok 10501, Thailand. E-mail: dung@cs.ait.ac.th.

Received March 1993; revised November 1993; accepted June 1994.

655 Avenue of the Americas, New York, NY 10010

0743-1066/95/$9.50
SSDI 0743-1066(94)00022-X

predicate symbol. Roughly, this completed definition is obtained by replacing the "if" by "iff." However, the Clark's completion does not always capture the intended meaning of logic programs. For example, let P consist of the single clause $p \leftarrow p$. Intuitively, we expect that any meaningful semantics of P would imply that p is false. But since comp(P) is $p \leftrightarrow p$, we cannot conclude from comp(P) that p is false. Let us consider one more example.

Example 1 [35]. Let P be

$$\begin{aligned} \text{edge}(a,b) &\leftarrow \\ \text{edge}(c,d) &\leftarrow \\ \text{reachable}(a) &\leftarrow \\ \text{reachable}(x) &\leftarrow \text{reachable}(y),\ \text{edge}(y,x) \\ \text{unreachable}(x) &\leftarrow \neg\text{reachable}(x) \end{aligned}$$

P can be illustrated by the following picture:

$$\begin{array}{ll} \ldots\ldots & a \curvearrowright b \\ & c \curvearrowright d \end{array}$$

We obviously expect vertices c, d to be unreachable, and indeed, Clark's semantics implies that c, d are unreachable, i.e.,

$$\begin{aligned} \text{comp}(P) &\vdash \text{unreachable}(c) \text{ and} \\ \text{comp}(P) &\vdash \text{unreachable}(d). \end{aligned}$$

Now, adding to P the clause edge (d, c) will result in a new program P' which is illustrated by the following picture:

$$\begin{array}{ll} \ldots\ldots & a \curvearrowright b \\ & c \circlearrowright d \end{array}$$

Although it still appears to be expected from the given information that c, d are unreachable, the Clark's semantics of P' could not imply that c, d are unreachable, i.e.,

$$\begin{aligned} \text{comp}(P') &\nvdash \text{unreachable}(c) \text{ and} \\ \text{comp}(P') &\nvdash \text{unreachable}(d) \end{aligned}$$

So it is necessary to find new ways for specifying the semantics of logic programs. Two approaches are proposed: The stable model semantics [20] and the well-founded model semantics [19].

The stable semantics of a program is defined by the set of its stable models. The problem of stable semantics is that it is not defined for every logic program, e.g., the program consisting of the only clause $p \leftarrow \neg\, p$, has no stable models. To illustrate the seriousness of this problem, let us consider one more example.

Example 2 (The Barber's paradox). "Beardland is a small city where the barber Noel shaves every citizen who does not shave himself.

Does Noel shaves the mayor Casanova?
Does Noel shave himself?"

The problem can be represented by a logic program consisting of the clauses

$$\text{shave}(\text{Noel}, t) \leftarrow \neg\text{shave}(t, t)$$
$$\text{mayor}(\text{Casanova}) \leftarrow$$

Despite the confusion about who shaves Noel, we expect that Noel shaves the mayor Casanova. But this program has no stable model, i.e., we could not conclude anything with respect to the stable semantics. □

The idea of well-founded semantics is negation as (possibly infinite) failure, i.e., the failure (possibly in infinitary) to prove a fact (a ground atom) to be true leads to the acceptance of this fact being false. Formally, the well-founded semantics is defined by the well-founded model which is defined as the least fixed point of a monotonic operator [19]. In contrast to the stable semantics, the well-founded semantics is defined for every logic program. Its major shortcoming is its inability to handle conclusions which can be reached only by "proof by cases." The following example illustrates this problem.

Example 3. Let P be

$$a \leftarrow \neg b$$
$$b \leftarrow \neg a$$
$$c \leftarrow a$$
$$c \leftarrow b$$

It is reasonable to expect that c holds. But with respect to the well-founded semantics, all a, b, c, are unknown. Note that in this case, the stable semantics provides the expected conclusions. □

The diversity of different approaches in semantics of negation suggests that there is probably not a unique intended semantics for logic programs. Which semantics should be used depends on concrete applications. To be able to choose the "right" semantics among different ones, *it is of great importance to understand the inherent relations between them.*

One of the well-known and simple approaches to nonmonotonic reasoning is abduction. In the simplest case, it has the form

From A and $A \leftarrow B$
infer B as a possible "explanation" of A.

Abduction has been the focus of intensive research lately [7, 6, 4, 8, 16, 14, 26, 24, 34]. The relationship between abduction and negation as failure has been studied lately by Eshghi and Kowalski [14], who have pointed out that by viewing negative literals in a logic program as abductive hypotheses, an abductive characterization of stable model semantics can be obtained. Eshghi and Kowalski [14] have also given an abductive procedure for computing abductive explanations. But they left

the question of what is the semantics of their procedure unanswered. Another open question is about the relations between abduction and other semantics (e.g., well-founded semantics) of logic programming.

The two major semantic intuitions for knowledge representation are the minimalism and maximalism, also known in the literature as skepticism and credulism, respectively. A skeptical reasoner refuses to draw conclusions in ambiguous situations where a credulous (belief-hungry) reasoner tries to conclude as much as possible [42]. In our framework, the semantics corresponding to these two intuitions are defined by the well-founded extension and the preferred extensions, respectively. While SLS-resolution (with the SLDNF-resolution as an approximation) [37, 33] has been recognized as an appropriate proof procedure for the minimalism semantics; it is still open what is the corresponding proof procedure for the maximalism semantics.

The goal of this paper is to study these open problems.

In Part 1 of this paper, we give a simple and intuitive declarative semantics for logic programming with "negative literals as abductive hypotheses." We show that the new semantics captures, generalizes, and unifies in a simple way the different semantic concepts (e.g., well-founded semantics, stable semantics, etc.) in logic programming. We also give sufficient conditions for the coincidence of different semantics. In Part 2, we show that Eshghi and Kowalski's abductive procedure provides the proof theory for the abductive semantics of logic programming. To demonstrate the practical applicability of this procedure, we apply it to solve a modified version of the well-known stable marriage problem.

LIST OF CONTENTS

PART I: DECLARATIVE SEMANTICS

The idea of abduction is that, to predict the expected observations from an incomplete knowledge base, the user supplies a set of hypotheses as a part of an

explanation to the expected observations. This explanation is considered as a logical theory based on a restricted set of possible hypotheses. An explanation can also be viewed as a scenario in which some goal is true. The user provides which hypotheses are acceptable in such scenarios [34].

In general, the theory of abductive reasoning is based on the notion of **abduction frameworks** [34, 14] defined as triples $\langle KB, IC, H\rangle$ where $\langle KB\rangle$ is a first-order theory representing the knowledge base, H is a set of first-order formulae representing the possible hypotheses, and IC is a set of first-order formulae representing the integrity constraints used to determine the admissible explanations.

Given an abduction framework $\langle KB, IC, H\rangle$, a set of hypotheses $E \subseteq H$ is an abductive solution for a query Q iff

$$KB \cup E \vdash Q \text{ and}$$
$$KB \cup E \text{ satisfies } IC$$

Thus, any theory for abductive reasoning has to provide answers to the following two questions:

What does "$KB \cup E$ satisfies IC" mean? (declarative semantics)
How can we compute the abductive solutions? (operational semantics)

Since our goal in this paper is to study the relations between abduction and logic programming, we restrict ourselves on a special class of abduction frameworks corresponding to logic programs.

We assume the existence of a fixed finite alphabet L, big enough to contain all constants, function symbols, and predicate symbols occurring in any program considered in this paper. The **Herbrand base** of L is denoted by ***HB***. A **logic program** is a set of clauses of the form $A \leftarrow L_1 \wedge \ldots \wedge L_n$ where A is an atom and L_is are literals. To define the class of abduction frameworks corresponding to logic programs, we introduce for each predicate symbol p contained in L a new predicate symbol **not-p** of the same arity. The new predicates are called **abducible predicates**. Atoms of the abducible predicates are called **abducible atoms**. Ground abducible atoms are called **hypotheses**. The set of all hypotheses is denoted by ***HY***. Atoms in HB are called **ordinary atoms**. For every ordinary atom $A = p(t_1, \ldots, t_n)$, **not-A** denotes the corresponding abducible atom not-$p(t_1, \ldots, t_n)$.

An **abductive program** over the language L is an abduction framework $\langle KB, IC, H\rangle$ such that

KB is a definite Horn theory over $L \cup \{\text{not-}p \mid p \text{ is a predicate symbol in } L\}$ with no abducible predicates appearing in the heads of its clauses.
$IC = \{\leftarrow p(x) \wedge \text{not-}p(x) \mid p \text{ is a predicate symbol in } L\}$[1]
$H = HY$

REMARK 1. Since the set of hypotheses and integrity constraints are fixed for all abductive programs over the fixed language L, we often write shortly KB for the abductive program $\langle KB, IC, HY\rangle$.

[1] All variables occurring in any clause of $KB \cup IC$ are universally quantified at the front of this clause.

REMARK 2. For the sake of convenience and without loss of generality, we assume that all programs considered in Part 1 are ground.

A logic program P is transformed into an abductive program $\mathbf{P}^*$ by replacing every negative literal $\neg p(t_1, \ldots, t_n)$ in each clause body by not-$p(t_1, \ldots, t_n)$. For example, let P be $\{p \leftarrow \neg q\}$. Then P^* is $\{p \leftarrow \text{not-}q\}$.

1.1. Acceptable Hypotheses and Preferred Extensions

The semantics of abductive programs is based on the notions of scenario and extension [34] recalled in the following definition.

Definition 1. A **scenario** of an abductive program KB is a first-order theory $KB \cup H$ where $H \subseteq HY$ such that $KB \cup H \cup IC$ is consistent.

An **extension** of an abductive program KB is a maximal (with respect to set inclusion) scenario of KB. □

For any set of hypotheses $H \subseteq HY$, let $\text{Con}(H, KB) = \{A \in HB \mid KB \cup H \vdash A\}$.

Lemma 0. Let KB be an abductive program, and let H be a set of hypotheses. Then $KB \cup H \cup IC$ is consistent iff $Con(H, KB) \cup H \cup IC$ is consistent.

PROOF. "$\Leftarrow$". Let $T = \text{Con}(H, KB)$. Then it is clear that $I = T \cup H$ is the least Herbrand model of $KB \cup \{\text{not-}A \leftarrow \mid \text{not-}A \in H\}$. Assume that $KB \cup H \cup IC$ is inconsistent. Then I is not a model of IC. That means that $I \cup IC$ is inconsistent. Contradiction !!" $\Rightarrow$" Obvious. □

In general, not every extension specifies an expected semantics of an abductive program. For example, let $KB = \{p \leftarrow \text{not-}q\}$. KB has two extensions $C_1 = KB \cup \{\text{not-}q\}, C_2 = KB \cup \{\text{not-}p\}$. But it is clear that only C_1 captures the expected semantics of KB.

The problem we are facing here is to determine those extensions, called preferred extensions, which specify the intended semantics of an abductive program. In other words, we have to specify when a hypothesis is acceptable.

Intuitively, it is clear that **a hypothesis is acceptable if there is no evidence to the contrary.** Let us take a closer look at this plausible rule.

It is clear that the contrary of a hypothesis not-A is the ordinary atom A. Hence, an evidence to the contrary of not-A can be considered as an evidence of A.

Definition 2. Let KB be an abductive program. A set of hypotheses $E \subseteq HY$ is called an **evidence** of an atom $A \in HB$ wrt KB if $KB \cup E \vdash A$. □

At first look, it seems appropriate to view the inconsistency of $E \cup S \cup IC$ for each evidence E of A as the formal interpretation of the condition that there exists no evidence to the contrary of not-A wrt a scenario S. But unfortunately, this cannot go well, as the following example shows.

Example 4. Let $KB : p \leftarrow \text{not-}p$. Let S be the scenario $KB \cup \emptyset$. The only evidence of p is $\{\text{not-}p\}$. It is obvious that $S \cup \{\text{not-}p\} \cup IC$ is inconsistent. Thus,

the hypothesis not-p would have no evidence to the contrary. So not-p should be acceptable. But it is clear that not-p could not be accepted since $S \cup \{\text{not-}p\} \cup IC$ is inconsistent. □

How could we interpret the condition "No Evidence to the Contrary" in the plausible rule?

We say that an evidence E of an atom A is defeated by a scenario S if there is not-$B \in E$ such that $S \vdash B$. So a hypothesis is acceptable iff each evidence to its contrary is defeated.

Definition 3. *A hypothesis not-A is said to be* **acceptable** *wrt a scenario S if for every evidence E of A, there is not-$B \in E$ such that $S \vdash B$.* □

It is clear that we are only interested in scenarios whose hypotheses are acceptable. Hence, the following definition.

Definition 4. *A scenario $S = KB \cup H$ is* **admissible** *if each hypothesis not-$A \in H$ is acceptable wrt S.* □

Definition 5. *A* **preferred extension** *of an abductive program KB is a maximal (wrt set inclusion) admissible scenario of KB.* □

Example 5. Let $KB = \{p \leftarrow \text{not-}q\}$. $S_1 = KB \cup \{\text{not-}q\}$. $S_2 = KB \cup \{\text{not-}p\}$ are two extensions of KB. Since q has no evidence, not-q is acceptable wrt S_1. So S_1 is admissible. Since $\{\text{not-}q\}$ is an evidence of p and $S_2 \nvdash q$, not-p is not acceptable wrt S_2. Thus, S_2 is not admissible. Hence, S_1 is the unique preferred extension of KB. □

Example 6. Let KB be $p \leftarrow \text{not-}p$. The only admissible scenario wrt KB is $KB \cup \emptyset$, which is also its unique preferred extension. □

Example 7 (Continuation of Example 2). Let us consider again the abductive program

$$\begin{aligned} \text{shave(Noel}, t) &\leftarrow \text{not-shave}(t, t) \\ \text{mayor(Casanova)} &\leftarrow \end{aligned}$$

It is not difficult to see that $S = KB \cup \{\text{not-shave}(c, c) \mid c \neq \text{Noel}\}$ is the only preferred extension. Thus, $S \vdash \text{shave(Noel}, c)$ for each $c \neq$ Noel. That means that our new semantics implies that Noel shaves every person except himself. Thus, Noel shaves the mayor Casanova. □

The following lemma shows the correctness of the above definitions.

Lemma 1 (Fundamental Lemma). *Let S be an admissible scenario, and let not-A, not-B be acceptable with respect to S. Then*

1) $S' = S \cup \{\text{not-}A\}$ is admissible
2) not-B is acceptable with respect to S'.

PROOF. 1) Let $S = KB \cup H$ and $H' = H \cup \{\text{not-}A\}$. We need only to prove that $H' \cup KB \cup IC$ is consistent. Assume the contrary. Hence, from Lemma 0, there is an atom B such that $KB \cup H' \vdash B$ and not-$B \in H'$. Thus, H' is an evidence of B. Since not-B is acceptable with respect to S, there is not -$X \in H'$ such that $S \vdash X$. Since $H \cup KB \cup IC$ is consistent and $H' = H \cup \{\text{not-}A\}$, it follows that $X = A$. Therefore, H is an evidence of A. Hence, there is not-$Z \in H$ such that $S \vdash Z$. It follows that $S \cup IC$ is inconsistent. Contradiction !! So $KB \cup H' \cup IC$ is consistent.

2) Obvious. □

Let us denote the set of all admissible scenarios of KB by $\boldsymbol{AS_{KB}}$. The existence of at least one preferred extension for every program KB is guaranteed by the following theorem.

Theorem 1. 1) $(AS_{KB}, \subseteq)$ is a complete partial order, i.e., every directed subset of AS_{KB} has a least upper bound. 2) For every admissible scenario S, there is at least one preferred extension K such that $S \subseteq K$.

PROOF. 1) Let Ω be a directed subset of AS_{KB}. Let $S = \cup\{S' \mid S' \in \Omega\}$. We want to show that S is an admissible scenario. First, we have to show that $S \cup IC$ is consistent. Assume the contrary. Hence, there exists an ordinary atom A s.t. $S \vdash A$ and not-$A \in S$. Therefore, there exists $S' \in \Omega$ s.t. $S' \vdash A$ and not-$A \in S'$. Contradiction. Now, we want to show that each hypothesis not-$A \in S$ is acceptable wrt S. Let E be an evidence of an arbitrary atom A with not-$A \in S$. From the definition of S, there is $S' \in \Omega$ such that not-$A \in S'$. Therefore, there is not-$B \in E$ such that $S' \vdash B$. Hence, $S \vdash B$. So S is an admissible scenario. It is clear that S is the least upper bound of Ω wrt set inclusion.

2) Obvious. □

1.2. Relations to Stable Semantics

Let KB be an abductive program.[2] A set $M \subseteq HB$ is called a *stable model* of KB iff M is the least Herbrand model of the program KB_M obtained by 1) deleting each clause in KB whose body contains an abducible atom not-A s.t. $A \in M$, and 2) deleting all abducible atoms in the bodies of the remaining clauses [20].

In the following, we introduce the notion of stable extension. This notion has been used by Eshghi and Kowalski [14] (under another name) to relate negation as failure to abduction.

A **stable extension** is a scenario S such that for every ordinary atom $A \in HB$, either $S \vdash A$ or not-$A \in S$ holds.

Eshghi and Kowalski [14] have proved the following lemma.

Lemma 2. Let KB be an abductive program and let M be a set of ground ordinary atoms. Then M is a stable model of KB if and only if there is a stable extension S of KB such that $M = \{A \mid A$ is an atom and $S \vdash A\}$. □

It is easy to see.

[2] See Remark 2.

Theorem 2. Every stable extension, is a preferred extension, but not vice versa. □

Theorem 1, 2 show that the preferred extension semantics generalizes stable semantics while overcoming its shortcoming. Now, it is interesting to ask the question as to under which conditions preferred extension and stable semantics coincide.

1.3. Coincidence Between Stable and Preferred Extension Semantics

Similar to the stable semantics, the Clark's completion semantics is not defined for every logic program. So much attention has been paid in the literature to find sufficient conditions for the consistency of Clark's predicate completion [27, 38]. Among the proposed conditions, Sato's order-consistency [38] is the most general one. Later, Dung and Fages [10, 17] have showed that order-consistency also guarantees the existence of at least one stable model. In this section, we will show that order-consistency is sufficient for the coincidence between preferred extension and stable semantics. Later, we will give also sufficient conditions for the coincidence among preferred extension, well-founded, and stable semantics.

The *atom dependency graph* of an abductive program KB is defined as follows. The nodes of the graph consist of ground ordinary atoms in HB. There is a positive (resp. negative) edge from an atom A to an atom B iff there is a ground clause in KB whose head is A and whose body contains B (resp. not-B).

The binary relations $\succeq_{+1}, \succeq_{-1}$ are defined as follows: $A \succeq_{+1} B$(resp. $A \succeq_{-1} B$) iff there is a path from A to B containing an even (resp. odd) number of negative edges in the dependency graph. Further define

$A \ggg B$ iff $A \succeq_{+1} B$ and $A \succeq_{-1} B$

$A \sqsupseteq B$ iff there is a path from$A$ to B containing at least one negative edge.

An abductive program is said to be *order-consistent* if $\ggg$ is a well-founded relation [38]. An abductive program is said to be *locally stratified* if $\sqsupseteq$ is well-founded [31].

It is not difficult to see that locally stratified programs are order-consistent, but not vice versa.

The following theorem shows that preferred extension semantics and stable semantics coincide for order-consistent programs.

Theorem 3. The preferred extensions of order-consistent programs are stable.

PROOF. See Appendix A. □

Example 8. Let KB :

$$P \leftarrow q$$
$$P \leftarrow r$$
$$r \leftarrow \text{not-}q$$
$$q \leftarrow \text{not-}r$$

$\ggg = \{(p,q),(p,r)\}$. It is clear $\ggg$ is well-founded. Hence, KB is order-consistent. KB indeed has exactly two preferred extensions which are stable. □

1.4. Well-Founded and Complete Extension Semantics

We will show in this section how the idea of well-founded semantics of logic programming can be captured in our framework.

Let us denote the *least admissible scenario* $KB \cup \emptyset$ of an abductive program KB by $\$_{KB}$.

An abductive program may have different semantics which can be defined as certain sets of admissible scenarios. An interesting question is whether or not there is a general characterization of these semantics. Since every semantics of an abductive program represents in some sense a possible world of the program, we expect that this world is complete in the sense that every acceptable hypothesis must be accepted. Thus, the class of scenarios in which all acceptable hypotheses are accepted is to special interest to us.

Definition 6. An admissible scenario S is called a **complete extension** *if every acceptable hypothesis wrt S is accepted in S.* In other words, an admissible scenario S is a complete extension if for any hypothesis not-A if not-A is acceptable wrt S then not-$A \in S$. □

From the definition of preferred extensions as maximal admissible scenarios, it follows immediately:

Theorem 4. Preferred extensions are complete extensions, but not vice versa. □

An example for the existence of a complete extension which is not a preferred extension is the least scenario of the program $KB = \{p \leftarrow \text{not-}q, q \leftarrow \text{not-}p\}$.

Let $\boldsymbol{CE_{KB}}$ be the class of all complete extensions of KB.

Each admissible scenario S has a **complete closure** which is the least (wrt set inclusion) complete extension containing S. We give now a construction for computing the complete closure of a scenario S.

Let $S \in AS_{KB}$. Define $\boldsymbol{V_{KB}} : \boldsymbol{AS_{KB}} \rightarrow \boldsymbol{AS_{KB}}$ by $V_{KB}(S) = S \cup \text{Acc}_{KB}(S)$ where $\text{Acc}_{KB}(S)$ is the set of all acceptable hypotheses wrt S.[3]

The complete closure of an admissible scenario S can be constructed as the limit of the following sequence $(S_i)_i$ of scenarios.

$$\begin{aligned} S_0 &= S \\ S_i &= \cup\{S_j \mid j < i\} \text{ for limit ordinal } i \\ S_{i+1} &= V_{KB}(S_i) \end{aligned}$$

It is not difficult to see that $(S_i)_i$ is an increasing sequence (wrt set inclusion). Hence, it has a limit $\boldsymbol{S^*}$ at some countable ordinal. From the fundamental lemma and Theorem 1, it follows immediately.

Lemma 3. S^* is the complete closure of S, i.e., S^* is a complete extension, and for every complete extension R if $S \subseteq R$, then $S^* \subseteq R$. □

Definition 7. The complete closure of the least admissible scenario is called the **well-founded extension** *denoted by* WFE (*i.e.*, $\boldsymbol{WFE_{KB}} = \$^*_{KB}$). □

[3]The correctness of the definition follows directly from the fundamental lemma.

Theorem 5. 1) $(CE_{KB}, \subseteq)$ *is a complete semilattice.*[4] 2) WFE_{KB} *is the least element of* $(CE_{KB}, \subseteq)$.

PROOF. 1) From Theorem 1 and Lemma 3, it follows immediately that $(CE_{KB}, \subseteq)$ is a complete partial order. We need only to show that every nonempty subset of CE_{KB} has an inf. Let R be a nonempty subset of complete extensions. Let $RS = \{S \in AS_{KB} \mid S \subseteq E \text{ for each } E \in R\}$. It is clear that RS is not empty and directed. Let $SE = \sup(RS)$. It is clear that $SE \in RS$. Hence, $SE^* \in RS$. Thus, $SE^* = SE$. So SE is clearly the infimum of R in $(CE_{KB}, \subseteq)$.

2) Obvious. □

The following theorem shows the coincidence between Van Gelder et al.'s well-founded model [19] and our well-founded extension.

Theorem 6. Let KB *be a logic program and* WFM_{KB} *be the well-founded model*[5] *of* KB. *Then*

$$WFM_{KB} = \{A \mid A \in HB \text{ and } WFE_{KB} \vdash A\} \cup \{\text{not-}A \mid \text{not-}A \in WFE_{KB}\}$$

PROOF. See Appendix B. □

From the coincidence between well-founded semantics and stable semantics for locally stratified programs [20, 19], it follows immediately that

Theorem 7. Locally stratified programs have exactly one preferred extension which is stable and well-founded. □

A more general class of programs for which the preferred extension and well-founded semantics coincide can be found in [10].

As a program may have different semantics (represented by some complete extensions) representing the different views peoples may draw from the program, it is meaningful to ask whether all of these different views may have something in common. From Theorem 5, it follows immediately that the well-founded semantics defined by well-founded extension represents the common ground for different semantics of a program. In other words, well-founded semantics is some kind of a skeptical semantics.

1.5. Relations to Other Approaches

To overcome the shortcoming of stable model semantics, Przymusinski has proposed the (three-valued) stable models [32] and the stationary semantics [36]. Sacca' and Zaniolo [40] have also introduced the partial stable models for the same purpose.

[4] A partial order $(R, \sqsubseteq)$ is a complete semilattice if every nonempty subset of R has an inf and every nonempty directed subset of R has a sup.

[5] See Appendix B for a formal definition of the well-founded model.

Although based on totally different concepts, it turns out that partial stable models are equivalent to preferred extensions [23]. Later, Brogi et al. [2] showed the equivalence between complete extensions and stationary expansions of logic programs (due to the fact that the well-founded model coincides with the least stationary expansion, this equivalence result is another proof for Theorem 6). Thus, we can say that our framework provides a simple and intuitive framework for semantics of logic programming unifying different intuitions and approaches.

Our approach is based on the notion of evidence defined in Definition 2. Kakas and Mancarella [22] argue that this notion of evidence is too strong in the sense that it views a set of hypotheses H satisfying the condition $KB \cup H \vdash A$, as an evidence of an atom A even if $KB \cup H \cup IC$ is inconsistent. Their idea can be illustrated by the following example:

Example. Let KB be

$$r \leftarrow \text{not-}p$$
$$p \leftarrow q$$
$$q \leftarrow \text{not-}q$$

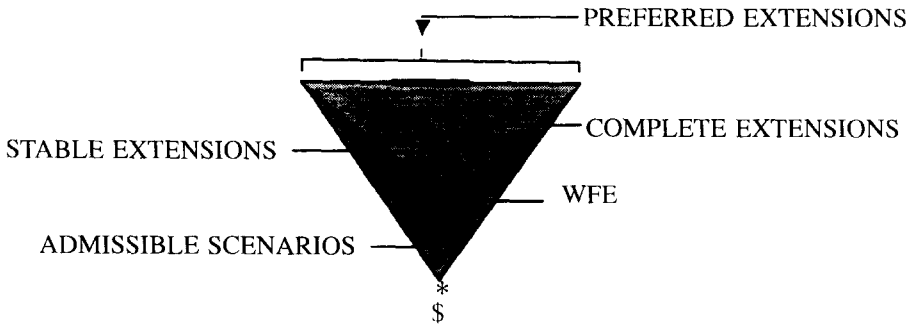
The only complete extension of KB is $KB \cup \emptyset$, i.e., nothing could be concluded from this program. But consider the scenario $KB \cup \{\text{not-}p\}$. The only could-be evidence to the contrary of not-p is $K = \{\text{not-}q\}$. But since $KB \cup K \cup IC$ is inconsistent, K should not be considered as evidence to the contrary of not-p. So $KB \cup \{\text{not-}p\}$ should be admissible. That means that r should be concluded in this example. □

Based on the above illustrated idea, Kakas and Mancarella [22] have proposed a modification of preferred extension semantics allowing more information to be drawn from a program than our semantics. Interesting results related to this idea can also be found in [41]. Later, Pereira et al. [30] further developed Kakas and Mancarella's idea and proposed the O-semantics. Further, Dung et al. [13, 12], and recently Alferes et al. [1], have demonstrated that the framework proposed in this paper can be generalized to provide a natural framework for the study of logic programming with "classical" negation. In another development, Kakas et al. [21] have pointed out that our abductive framework embodies in fact an argumentational approach to semantics of logic programming. This is a fundamental insight. Generalizing this idea, in a recent work [11], we developed a simple and general theory for argumentation, and showed that logic programming as well as nonmonotonic reasoning are just different forms of argumentation. A simple general and unifying argument-based framework for nonmonotonic reasoning has also been developed recently by Bondarenko et al. [3]. These new works give a qualitative new insight into the nature of logic programming and nonmonotonic reasoning.

SUMMARY

The results in Part 1 can be illustrated in the following picture:

We can say that the set of complete extensions, CE_{KB}, represents *the universe of possible semantics* of an abductive program in which well-founded semantics corresponds to the "minimalism" semantics where only things which hold in all possible



worlds are "believed," while the preferred extension semantics corresponds to the "maximalism" semantics where each preferred extension represents a "belief world" of an agent who tries to conclude as much knowledge as possible from an abductive program considered as an incomplete knowledge base. In that sense, complete extensions represent a "moderate" point of view which somehow lies between the two extreme positions of minimalism and maximalism.

PART II: COMPUTING ABDUCTIVE SOLUTIONS

A set of hypotheses E is called an **abductive solution** for a query Q wrt an abductive program KB iff

$$KB \cup E \vdash Q \text{ and}$$
$$KB \cup E \text{ is admissible.}$$

It is clear that if there exists an abductive solutions E for Q wrt KB, then there exists a preferred extension $KB \cup H$ of KB such that $E \subseteq H$. Hence, any procedure for computing abductive solution is also a proof procedure for preferred extension semantics.

The main goal of this part is to show that the procedure given by Eshghi and Kowalski in [14] is a procedure for computing abductive solutions.

From now on, we call Eshghi and Kowalski's abductive procedure simply the EK-procedure for short.

2.1. Eshghi and Kowalski's Abductive Procedure

The abductive procedure can be viewed as an extension of the SLDNF-resolution consisting of two interleaving activities: (a) reasoning backward for a refutation and collecting the required hypotheses, as shown inside an ordinary box in Example 9, and (b) checking that the collected hypotheses are consistent, as shown in the bold boxes in Example 9.

Let KB be an abductive program. Let R be a safe computation rule (one that selects an abducible atom only if it is ground).

An **abductive derivation** from (G_1, H_1) to (G_n, H_n) is a sequence

$$(G_1, H_1), (G_2, H_2), \ldots, (G_n, H_n)$$

such that, for each $i, 0 < i < n, G_i$ has the form $\leftarrow l, l'$ where (without loss of generality) R selects l, and l' is a (possibly empty) collection of atoms, H_i is a set of hypotheses, and

ab1) if l is not abducible

$$\text{then} \quad G_{i+1} = C \quad \text{and} \quad H_{i+1} = H_i$$

where C is the resolvent of some clause in KB with the clause G_i on the selected literal l.

ab2) If l abducible and $l \in H_i$

$$\text{then} \quad G_{i+1} = \leftarrow l' \text{ and } H_{i+1} = H_i$$

ab3) If l is abducible (l = not-k) and $l \notin H_i$ and there is a consistency derivation from $(\{\leftarrow k\}, H_i \cup \{l\})$ to $(\emptyset, H')$

$$\text{then } G_{i+1} = \leftarrow l' \quad \text{and} \quad H_{i+1} = H'$$

An **abductive refutation** is an abductive derivation to a pair $(\square, H')$.

A **consistency derivation** from (F_1, H_1) to (F_n, H_n) is a sequence

$$(F_1, H_1), (F_2, H_2), \ldots, (F_n, H_n)$$

such that, for each $i, 0 < i < n, F_i$ has the form $\{\leftarrow l, l'\} \cup F_i'$, where (without loss of generality) the clause $\leftarrow l, l'$ has been selected (to continue the search), R selects l, and

co1) If l is not abducible

$$\text{then} \quad F_{i+1} = C' \cup F_i' \quad \text{and} \quad H_{i+1} = H_i$$

where C' is the set of all resolvents of clauses in KB with the selected clause on the selected literal, and $\square \notin C'$.

co2) If l is abducible, $l \in H_i$ and l' is not empty

$$\text{then} \quad F_{i+1} = \{\leftarrow l'\} \cup F_i' \quad \text{and} \quad H_{i+1} = H_i$$

co3) If l is abducible (l = not-k) and $l \notin H_i$
then if there is an abductive derivation from $(\leftarrow k, H_i)$ to $(\square, H')$

$$\text{then} \quad F_{i+1} = F_i' \quad \text{and} \quad H_{i+1} = H'$$

else if l' is not empty

$$\text{then } F_{i+1} = \{\leftarrow l'\} \cup F_i' \quad \text{and} \quad H_{i+1} = H_i.^6$$

Example 9. Let KB be the following program:

$$p \leftarrow \text{not-}q$$
$$q \leftarrow \text{not-}p$$

[6] In the original definition of this procedure [14], $H_{i+1} = H_i \cup \{l\}$, which is a minor error. See Appendix D for more details.

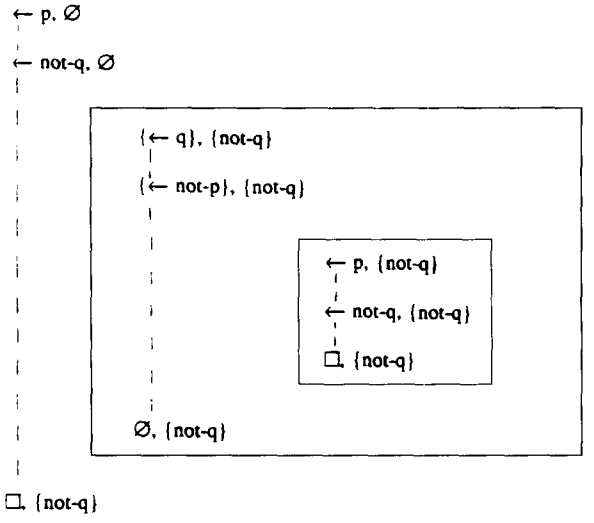


FIGURE 1

The search space for the goal $(\leftarrow p, \emptyset)$ is given in Figure 1.□

The correctness of the abduction procedure for an abductive program KB means that whenever there exists an abductive derivation from $(\leftarrow A, \emptyset)$ to $(\square, H)$ for $A \in HB$, then H is an abductive solution of A.

Theorem 8 (Soundness of the abductive procedure). *Let* $(\leftarrow A, \emptyset), \ldots, (\square, H)$ *be an abductive refutation. Then* $KB \cup H$ *is an admissible scenario and* $KB \cup H \vdash A$.

PROOF. See Appendix C. □

In general, the EK-procedure given above is not complete, as the following example shows.

Let KB be the following program:

$$p \leftarrow \text{not-}q$$
$$q \leftarrow q$$

It is clear that $E = \{\text{not-}q\}$ is an abductive solution for p, but there is no abductive refutation for the goal $\leftarrow p$ (due to the loop caused by the second clause).

To have a complete procedure, several problems have to be addressed. First, a mechanism for loop checking is needed. Second, a form of constructive negation is needed to overcome the floundering problem. The second question has been addressed in a recent paper of Kakas and Mancarella [25].

Another kind of completeness of the EK-procedure has been recently studied by Giordano et al. [18]. Instead of extending the EK-procedure to provide a complete procedure for computing abductive solutions, they have changed the semantics to fit the EK-procedure. A three-valued semantics has been proposed, and the completeness of the EK-procedure with respect to this semantics has been shown for ground programs [18].

2.2. EK-Procedure and the Stable Marriage Problem

The stable marriage problem is a special case of the graph matching problem which has been studied extensively in the literature due to its wide applicability[7] [39]. In [29], Marek et al. have shown that there is a close relation between the stable marriage problem and nonmonotonic reasoning. Later, in [11], we show that the stable marriage problem can best and most naturally be viewed as a problem of argumentation. In this section, we will demonstrate how the EK-procedure can be applied to solve a modified version of the stable marriage problem in a natural way.

Assume that we have N men and M women who have expressed mutual preference (each man must say how he feels about each woman and vice versa). Let A be one of the men and let B be one of the women. Let us imagine further that, due to some hidden reason, you want to arrange a marriage between A and B. Now, the problem is that A may prefer someone to B or B may prefer someone to A. For example, A may prefer C to B. So to prevent A from running away with C, you would have to arrange a marriage for C with someone whom she prefers to A. In short, to create a stable marriage between A and B, you have to create a stable marriage for all those whom A (resp. B.) prefers to B (resp. A).

The above problem can be represented by the following logic program $P = P1 \cup P2$ with

$$P1 : \text{smarriage}(X,Y) \leftarrow \neg\ \text{umarriage}(X,Y)$$
$$\text{umarriage}(X,Y) \leftarrow\ \text{prefer}(X,Z,Y),\ \text{smarriage}(X,Z)$$
$$\text{umarriage}(X,Y) \leftarrow\ \text{prefer}(Y,Z,X),\ \text{smarriage}(Z,Y)$$

$P2$: a set of facts about the preference of the involved individuals

where smarriage(X,Y) means that a marriage between X,Y is stable, umarriage (X,Y) means that a marriage between X,Y is unstable, and prefer(X,Z,Y) means that X prefers Z to Y.

Let KB be the corresponding abductive program of P. It is not difficult to see that the task of arranging a stable marriage between A and B can be reduced to the task of finding an admissible scenario $KB \cup H$ of KB s.t. $KB \cup H \vdash$ smarriage (A,B). Such a scenario can be found by applying the EK-procedure to the query $\leftarrow$smarriage(A,B). Let assume, for example, that we have three men and three women with the following lists of preference:

A	B	C		a	b	c
b	a	b		A	B	A
a	b	c		B	A	C
c	c	a		C	C	B

We want to arrange a marriage between A and b. Since b prefers B to A, we have to get B married to someone whom he/she prefers to b to avoid b running away with B. Since B prefers a to everybody, it seems the best solution is to get a to agree to marry B. a prefers only A to B, but A prefers b to a, so there is not chance for

[7]For example, in the U.S., a quite complicated system has been set up to place graduating medical students into hospital residency positions. Each student lists several hospitals in order of preference, and each hospital lists several students in order of preference. The problem is to assign the students to positions in a fair way, respecting all the stated preferences.

a to get A, and since B is the second best choice for a, it is clear that a agrees to marry B. Thus, to make the marriage between A and b stable, we have helped B marry a and this marriage is also stable.

The corresponding abductive program is

$$\begin{aligned}
\text{smarriage}(X,Y) &\leftarrow \text{not-umarriage}(X,Y)\\
\text{umarriage}(X,Y) &\leftarrow \text{prefer}(X,Z,Y),\ \text{smarriage}(X,Z)\\
\text{umarriage}(X,Y) &\leftarrow \text{prefer}(Y,Z,X),\ \text{smarriage}(Z,Y)
\end{aligned}$$

$$\begin{aligned}
\text{prefer}(A,b,a) &\leftarrow\\
\text{prefer}(A,a,c) &\leftarrow\\
\text{prefer}(A,b,c) &\leftarrow
\end{aligned}$$

$$\begin{aligned}
\text{prefer}(B,a,b) &\leftarrow\\
\text{prefer}(B,b,c) &\leftarrow\\
\text{prefer}(B,a,c) &\leftarrow\\
&\cdots\cdots\\
&\cdots\cdots\\
\text{prefer}(a,A,B) &\leftarrow\\
\text{prefer}(a,B,C) &\leftarrow\\
\text{prefer}(a,A,C) &\leftarrow
\end{aligned}$$

$$\begin{aligned}
\text{prefer}(b,B,A) &\leftarrow\\
\text{prefer}(b,A,C) &\leftarrow\\
\text{prefer}(b,B,C) &\leftarrow\\
&\cdots\cdots\\
&\cdots\cdots
\end{aligned}$$

The search space for the query $\leftarrow \text{smarriage}(A,b)$ is given in Figure 2.[8]

CONCLUSION

We have shown that the abductive framework provide a simple basis for declarative and operational semantics of logic programs. We have introduced the notions of admissible scenarios, preferred, and complete extensions, and have demonstrated how these new notions provide a unified framework which captures and generalizes different semantic concepts (e.g., well-founded models, stable models) in logic programming. The key step is the way we interpret the plausible rule that a hypothesis is acceptable if there is no evidence to the contrary.

We argue that, in general, well-founded semantics is a minimalism semantics and preferred extension semantics is a maximalism semantics, while complete extensions

[8]Due to space constraints, we write sm for smarriage and um for umarriage. Further, we often omit to writing down explicitly the set of hypotheses H_{i+1} in a derivation step if $H_i = H_{i+1}$.

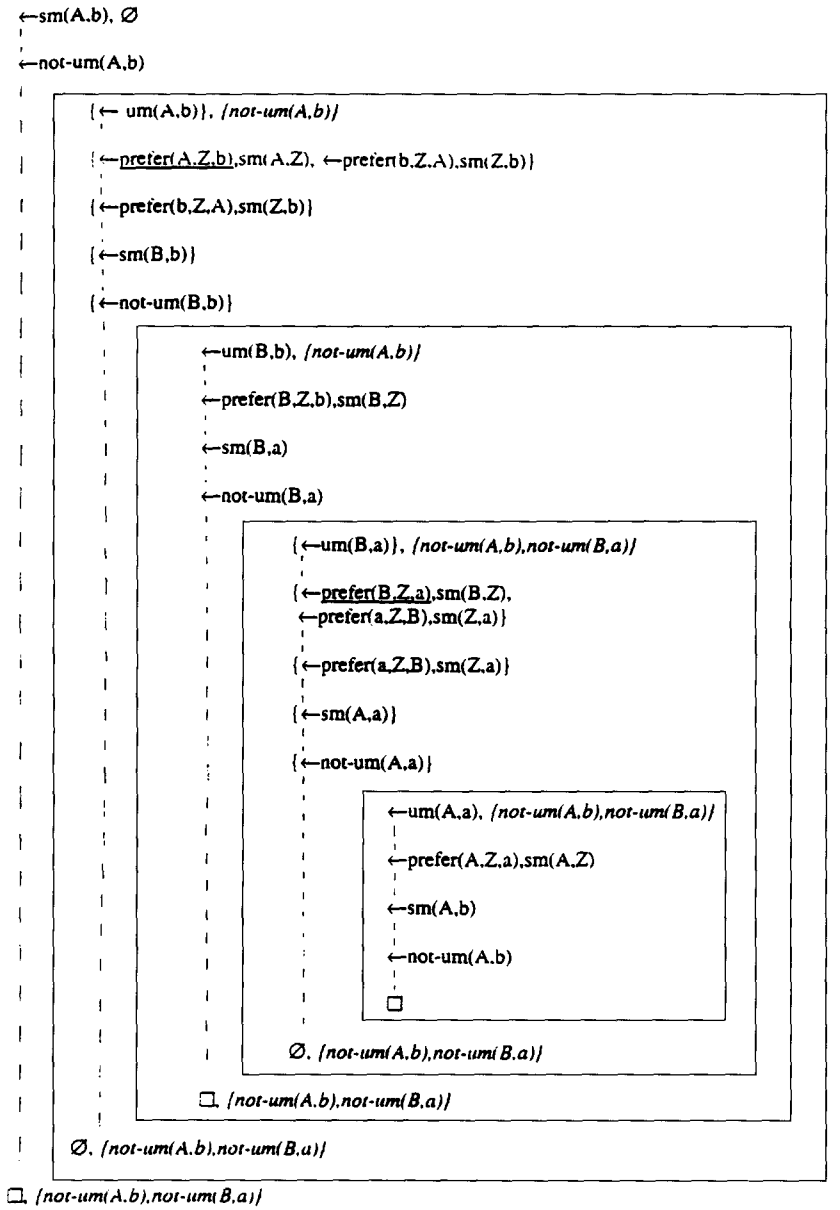

←sm(A,b), ∅
←not-um(A,b)
{← um(A,b)}, {not-um(A,b)}
{←prefer(A,Z,b),sm(A,Z), ←prefer(b,Z,A),sm(Z,b)}
{←prefer(b,Z,A),sm(Z,b)}
{←sm(B,b)}
{←not-um(B,b)}
←um(B,b), {not-um(A,b)}
←prefer(B,Z,b),sm(B,Z)
←sm(B,a)
←not-um(B,a)
{←um(B,a)}, {not-um(A,b),not-um(B,a)}
{←prefer(B,Z,a),sm(B,Z),
←prefer(a,Z,B),sm(Z,a)}
{←prefer(a,Z,B),sm(Z,a)}
{←sm(A,a)}
{←not-um(A,a)}
←um(A,a), {not-um(A,b),not-um(B,a)}
←prefer(A,Z,a),sm(A,Z)
←sm(A,b)
←not-um(A,b)
□
∅, {not-um(A,b),not-um(B,a)}
□, {not-um(A,b),not-um(B,a)}
∅, {not-um(A,b),not-um(B,a)}
□, {not-um(A,b),not-um(B,a)}

FIGURE 2

form a kind of a "moderate" semantics. Further, we have shown that for the class of order-consistent programs, preferred extensions and stable semantics coincide.

APPENDIX A

A set $I \subseteq HB \cup HY$ of ground ordinary and abducible atoms is consistent if there is no atom A s.t. both A and not-A belong to I. In [10, 17], the following proposition is proved:

Proposition 1. If KB is an order-consistent program, then KB has at least one stable extension. □

Proposition 2. Let KB be an arbitrary abductive program, and let $S = KB \cup H$ be an arbitrary preferred extension of KB. Further, let C be a clause with head A in KB such that not-$A \in H$. Then body$(C) \cup H$ is inconsistent.

PROOF. Assume the contrary. So each ordinary atom $B \in$ body(C) has an evidence E_B such that for each not-$X \in E_B, S \nvdash X$. Let $E = \cup\{E_B \mid B \in$ body$(C) \cap HB\} \cup$(body$(C) \cap HY)$. E is an evidence of A. Since not-$A \in H$, there is not-$B \in E$ such that $S \vdash B$. Contradiction !! □

Theorem 3. The preferred extensions of order-consistent programs are stable.

PROOF. Let $S = KB \cup H$ be an arbitrary preferred extension of KB. Let Q be the program obtained from KB by 1) deleting each clause C s.t. body$(C) \cup H$ is inconsistent, and 2) deleting all occurrences of abducible atoms not-$B \in H$ in the remaining clauses. It follows immediately that Q is again an order-consistent program. From Proposition 1, Q has at least one stable extension. From Proposition 2, it follows that none of the hypotheses in H appears in Q and for no ordinary atom A appearing in Q, not-A appears in H. Let $S' = Q \cup H'$ be a stable extension of Q. It is clear that H is a subset of H'. We want to show that $KB \cup H'$ is a stable extension of KB. First, from the consistency of $Q \cup H' \cup IC$ immediately follows the consistency of $KB \cup H' \cup IC$. Further, it is clear that for each $A \in HB$, either $KB \cup H' \vdash A$ or not-$A \in H'$. In other words, $KB \cup H'$ is stable. From the assumption that $KB \cup H$ is a preferred extension, it follows that $H = H'$. So S is a stable extension of KB. □

APPENDIX B

A set $I \subseteq HB \cup HY$ of ground ordinary and abducible atoms is consistent if there is no atom A s.t. both A and not-A belong to I. A partial interpretation is a consistent set of ordinary and abducible atoms. Given a partial interpretation I, a set S of ground ordinary atoms is called an *unfounded set* of an abductive program KB wrt I iff for each atom $A \in S$, for each clause C in KB whose head is A,[9] one

[9]Note that we assume that KB is ground.

of the following conditions holds:

1) body$(C) \cup I$ is inconsistent
2) There is at least one ordinary atom B in the body of C such that $B \in S$.

It is clear that the union of unfounded sets is again unfounded. Hence, there always exists a greatest unfounded set wrt partial interpretation I. This set is denoted by $GU(I)$.

Define

$$T_{KB}(I) = \{A \mid \exists C \in KB \text{ s.t. } A = \text{head } (C) \text{ and the body of } C \text{ is true wrt } I\}$$
$$W_{KB}(I) = T_{KB}(I) \cup \{\text{not-}A \mid A \in GU(I)\}$$

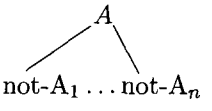
The well-founded model of KB, denoted by WFM_{KB}, is defined as the least fixed point of the monotonic operator W_{KB}[19].

For the proof of the coincidence between well-founded model and well-founded extension, we need the following notion of proof trees.

1) If $A \leftarrow$ is a clause in KB, then the tree

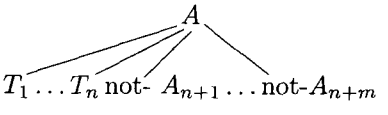
is a proof tree of A.

2) If $A \leftarrow \text{not-}A_1, \ldots, \text{not-}A_n$ is a clause in KB, then the tree



is a proof tree of A.

3) If $A \leftarrow A_1, \ldots, A_n$, not-$A_{n+1}, \ldots,$ not-A_{n+m} is a clause in KB, and $T_1, \ldots, T_n$ are proof trees of $A_1, \ldots, A_n$, respectively, then the tree



is a proof tree of A.

Lemma 4. Let KB be an abductive program, and let $S = KB \cup H$ be an admissible scenario of KB. Further, let I be a partial interpretation defined by $I = \text{Con}(H, KB) \cup H$. Then for each ordinary atom $A \in HB$, not-A is acceptable wrt S iff $A \in GU(I)$.

PROOF "$\Leftarrow$" Let $A \in GU(I)$. Assume that not-A is not acceptable wrt S, i.e., there is an evidence E of A such that for each not-$B \in E$, $S \not\vdash B$. Then there exists a proof tree Tr of A wrt KB whose terminal nodes belong to $E \cup \{\square\}$. We first prove the following proposition.

Proposition. There is a path in Tr from the root A to a terminal node not-B such that all the positive literals on this path belong to $GU(I)$ and $B \in I$.

PROOF. By induction on the height (the length of the longest path from the root to a terminal node) of Tr.

Base case: The height of Tr is 1. The proposition follows directly from the fact that $GU(I)$ is an unfounded set.

Induction case: Let the height of Tr be n. Let C be the clause such that body (C) is the set of all children of A in Tr. Since $GU(I)$ is an unfounded set and $A \in GU(I)$, it follows that either $I \cup$ body(C) is inconsistent or there is $B \in$ body$(C) \cap GU(I)$.

Case 1: $I \cup$ body(C) is inconsistent. Case 1.1: There is an ordinary atom $B \in$ body(C) such that not-$B \in I$. Hence, not-B is acceptable wrt S. As E is an evidence of B, there is not-B' in E such that $S \vdash B'$. Contradiction. So Case 1.1 does not occur. Case 1.2: There is an abducible atom not-B in body(C) such that $B \in I$. This leads to a contradiction since not-$B \in E$. So Case 1.2 cannot occur either.

Case 2: There is $B \in$ body$(C) \cap GU(I)$. The subtree Tr' with root B of Tr is again a proof tree of B wrt KB. As height of Tr' is less than or equal to $n-1$ and $B \in GU(I)$, it follows that there is a path from the root B to a terminal node not-B' in Tr' such that all the positive literals on this path belong to $GU(I)$ and $B' \in I$. The proposition then follows immediately.

The proposition implies immediately that for each $A \in GU(I)$, not-A is acceptable wrt S.

"$\Rightarrow$" Let $X = \{B \mid$ not-B is acceptable wrt $S\}$. We want to prove that X is an unfounded set of KB wrt I. Assume that X is not an unfounded set wrt I. Then there is an atom $A \in X$ and a clause $A \leftarrow Bd$ in KB such that $I \cup Bd$ is consistent and no ordinary subgoal in Bd belongs to X. Thus, there exists an evidence E_B for every ordinary subgoal B in Bd such that $S \not\vdash B'$ for each not-$B' \in E_B$. Let $E = \cup\{E_B \mid B \in Bd \cap HB\} \cup \{$not-$B \mid$ not-$B \in Bd\}$. Then it is clear that E is an evidence of A. Since not-A is acceptable wrt S, there is not-$B' \in E$ such that $S \vdash B'$. Thus, not-$B' \in Bd$. This is a contradiction to the fact that $I \cup Bd$ is consistent !! So X is unfounded wrt I. □

Theorem 6. Let KB be a logic program, and let WFM_{KB} be the well-founded model of KB. Then

$$WFM_{KB} = \{A \mid A \in HB \text{ and } WFE_{KB} \vdash A\} \cup \{\text{not-}A \mid \text{not-}A \in WFE_{KB}\}$$

PROOF. Follows immediately from Lemma 4. □

APPENDIX C

We show now the correctness of the abductive procedure.

Let

$\beta : (G_1, H_1), (G_2, H_2), \ldots, (\square, H_n)$ be an abductive refutation, and

$\mu : (G_1, K_1), (F_2, K_2), \ldots, (\emptyset, K_m)$ be a consistency derivation.

Define

$\beta > \mu$ if there is $G_i = \leftarrow l, l'$ such that $l = \text{not-}k, l \notin H_i$, and μ is a consistency derivation from $(\{\leftarrow k\}, H_i \cup \{l\})$ to $(\emptyset, H')$ such that $G_{i+1} = \leftarrow l'$ and $H_{i+1} = H'$

$\mu > \beta$ if there is $F_i = \{\leftarrow l, l'\} \cup F_i'$ such that $l = \text{not-}k, l \notin K_i$ and β is an abductive derivation from $(\leftarrow k, K_i)$ to $(\square, K')$ such that $F_{i+1} = F_i'$ and $K_{i+1} = K'$.

If $G_1 = \leftarrow A$, then the set of all hypotheses appearing in goals G_is in β is called the **evidence of A generated by** β.

Further, let $\gg$ be the transitive closure of $>$.
It is obvious that the following Lemmas hold.

Lemma 5.

1) *Let β be an abductive refutation, and let $\mu : (\{\leftarrow B\}, K), \ldots, (\emptyset, K')$ be a consistency derivation such that $\beta \gg \mu$. Then not-$B \in K$.*
2) *Let β be an abductive derivation and $\beta : (\leftarrow B, M), \ldots, (\square, M')$ be an abductive derivation such that $\beta \gg \beta$. Then not-$B \notin M$.*
3) *Let μ be a consistency derivation from $(\{\leftarrow A\}, K)$ to $(\emptyset, K')$. Let E be an evidence of A. Then for some not-$B \in E$, there exists an abductive refutation β from $(\leftarrow B, H)$ to $(\square, H')$ such that $\mu > \beta$.* □

Lemma 6. If there exists an abductive refutation from $(\leftarrow A, H)$ to $(\square, H')$, then there exists no consistency derivation from $(\{\leftarrow A\}, K)$ to $(\emptyset, K')$ for any K with $H' \subseteq K$. □

Lemma 7. Let β be an abductive refutation from $(\leftarrow A, H)$ to $(\square, H')$ such that not-$A \notin H$. Then not-$A \notin H'$.

PROOF. Assume the contrary. Then there exists a consistency derivation $\mu : (\{\leftarrow A\}, K), \ldots, (\emptyset, K')$ for some K such that $\beta \gg \mu$. Let E be the evidence of A generated by β. Thus, there exists an abductive refutation $\beta' : (\leftarrow B, M), \ldots, (\square, M')$ for some not-$B \in E$ such that $\mu > \beta'$ (Lemma 5.3). It is clear that not-$B \notin M$(Lemma 5.2). Since not-$B \in E$ and E is generated by β, there is a consistency derivation μ' from $(\{\leftarrow B\}, R)$ to $(\emptyset, R')$ such that $\beta > \mu'$. It is clear that not-$B \in R$. Since not-$B \notin M$ and not-$B \in R$, it follows that μ' (as a process) starts after β' in the process of β. Then either $\beta' \gg \mu'$ or β', μ' are disjoint. Since $\beta > \mu'$ and $\beta \gg \mu > \beta'$, it follows immediately that β', μ' are disjoint. Thus, $M' \subseteq R$. Lemma 6 implies that μ' does not exist. Contradiction!! □

Lemma 8. Assume that there is an abductive refutation β from $(\leftarrow A, \emptyset)$ to $(\square, H)$. Then $H \cup KB \cup IC$ is consistent.

PROOF From Lemma 5.3, it follows immediately

Proposition. Let not-$B \in H$ and E is an evidence of B. Then for some hypothesis

not-$X \in E$, there exists an abductive refutation β' from $(\leftarrow X, R)$ to $(\Box, R')$ such that $\beta \gg \beta'$.

Assume that $KB \cup H \cup IC$ is inconsistent. Then there is an atom X such that not-$X \in H$ and $X \in \text{Con}(H, KB)$. From not-$X \in H$ and H is an evidence of X, it follows from the above proposition that there is not-$B \in H$ with an abductive refutation β' from $(\rightarrow B, K)$ to $(\Box, K')$ such that $\beta \gg \beta'$. It is clear that not-$B \notin K$ (Lemma $C1.2$). But since not-$B \in H$, there is a consistency derivation μ from $(\{\leftarrow B\}, R)$ to (ϕ, R') such that $\beta \gg \mu$ and not-$\beta \in R$. There are two cases:

Case 1. $\beta' \gg \mu$. That means that not-$B \in K'$. Contradiction to Lemma 7.

Case 2. μ and β' are disjoint. Thus, μ (as a process) starts after β' (as a process) terminates in β (as a process). Then it is clear that $K' \subseteq R$. Lemma 6 implies that μ does not exist. Contradiction !! □

It follows immediately from Lemmas 5, 8.

Theorem 8 (Soundness of the abduction procedure). *Let $(\leftarrow A, \emptyset), \ldots, (\Box, H)$ be an abductive refutation. Then $KB \cup H$ is an admissible scenario such that $KB \cup H \vdash A$.* □

APPENDIX D: THE ORIGINAL ESHGHI AND KOWALSKI ABDUCTION PROCEDURE [14]

The definition of the original abduction procedure is similar to the definition of the modified abduction procedure given above, with the one exception that step (co3) is replaced by (co3′) as follows.

co3′) If l is a hypothesis $(l = \text{not-}k), l \notin H_i$
then if there is an abductive derivative from $(\leftarrow k, H_i)$ to $(\Box, H')$

$$\text{then} \quad F_{i+1} = F'_i \quad \text{and} \quad H_{i+1} = H'$$

else if l' is not empty

$$\text{then} \quad F_{i+1} = \{\leftarrow l'\} \cup F'_i \quad \text{and} \quad \mathbf{H_{i+1}} = \mathbf{H_i} \cup \{\mathbf{l}\} \quad \Box$$

The following theorem is given in [14].

Theorem. *Let KB be a locally stratified abductive program. Then the abduction procedure for KB is correct.* □

The following simple example shows that this is not the case.

COUNTEREXAMPLE. Let KB be the following program:

$$\begin{aligned} a &\leftarrow \text{not-}b \\ b &\leftarrow \text{not-}c \wedge \text{not-}r \\ r &\leftarrow \text{not-}c \\ c &\leftarrow \text{not-}d \\ d &\leftarrow d \end{aligned}$$

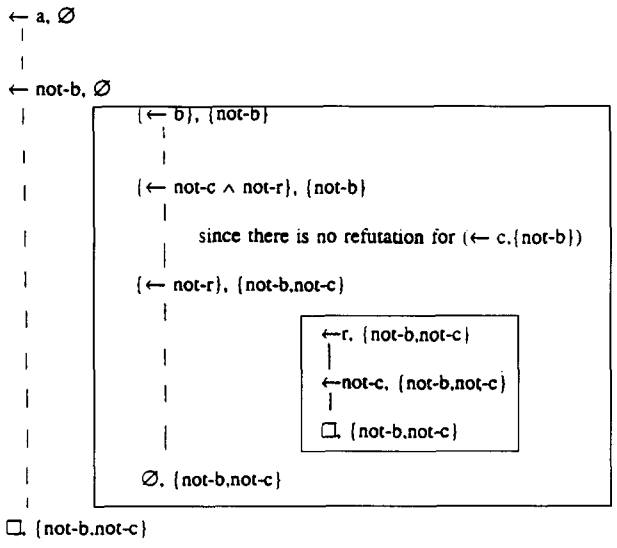


FIGURE 3

It is clear that KB is stratified, hence also locally stratified. The unique stable model of KB is $\{c, a\}$. The unique preferred extension of KB is $Æ = KB \cup H$ with $H = \{\text{not-}d,\ \text{not-}r,\ \text{not-}b\}$ and $\text{Con}(H, KB) = \{c, a\}$. Thus Æ is a stable extension.

Since there is not consistency derivation from $(\{\leftarrow d\}, \{\ \text{not-}b,\ \text{not-}d\})$ to $(\emptyset, S)$ for any S, there exists no abductive refutation from $(\leftarrow c, \{\text{not-}b\})$ to $(\Box, R)$ for any R. Thus, we obtain the search space shown in Figure 3.

If the above theorem is correct, then **not-c** $\in$ **{not-d, not−r, not-b} Contradiction!!**

So this example represents a counterargument to the above theorem that the abduction procedure is sound for locally stratified programs.

It is interesting to note that in an earlier and unpublished version of their abductive procedure, Eshghi and Kowalski [15] have used the following co3") instead of the above co3′.

co3″) If l is a hypotheses ($l = \text{not-}k$), $l \notin H_i$
then if there is an abductive derivation from $(\leftarrow k, H_i)$ to $(\Box, H')$

$$\text{then } F_{i+1} = F'_i \quad \text{and} \quad H_{i+1} = H'$$

else if l' is not empty, and there is no such derivation (the abductive proof procedure finitely fails to find one)

$$\text{then } F_{i+1} = \{\leftarrow l'\} \cup F' \quad \text{and} \quad \mathbf{H_{i+1} = H_i \cup \{l\}} \quad \Box$$

It is easy to see that this version of the abductive procedure works correctly for the above example. But a formal proof of its correctness remains to be found.

I am grateful to Robert Kowalski for his insightful discussions as well as his many valuable comments on an earlier version of this paper. Many thanks also to Tony Kakas, Paolo Mancarella, Mark Wallace, and Wolfgang Wechler for their comments and suggestions. I am also thankful to Huynh Ngoc Phien for his support during the time in which this paper was written. Last but not least, my sincere thanks to two anonymous referees for their constructive criticism.

REFERENCES

1. Alferes, J. J., Dung, P. M., and Pereira, L. M., Scenario Semantics of Extended Logic Programs, in: Pereira, L. M. and Nerode, A. (eds.), *Proceedings of the Second International Workshop on LPNMR*, MIT Press, Lisbon, 1993.
2. Brogi, A., Lamma, E., Mancarella, P., and Mello, P., Normal Logic Programs as Open Positive Programs, in Apt, K. (ed.), *Proceedings of the Joint International Conference and Symposium on Logic Programming*, MIT Press, Washington, 1992.
3. Bondarenko, A., Toni, F., and Kowalski, R., An Assumption-Based Framework for Nonmonotonic Reasoning, in: Pereira, L. M. and Nerode, A. (eds.), *Proceedings of the Second International Workshop on Logic Programming and Nonmonotonic Reasoning*, MIT Press, Lisbon, 1993.
4. Console, L., Dupre, D. T., and Torasso, P., On the Relationship Between Abduction and Deduction, *J. Logic Computat.* 1(5):661–690 (1991).
5. Clark, K. L., Negation as Failure, in: Gallaire, H. and Minker, J. (eds.), *Logic and Database*, Plenum, New York, 1978.
6. Cox, P. T. and Pietzykowski, T., Causes for Events: Their Computation and Application, in: *Proceedings of 8th International Conference on Automated Deduction, CADE-86,* Springer-Verlag, 1986, pp. 608–621.
7. Charniak, E. and McDermott, D., *Introduction to AI*, Addison-Wesley, Menlo Park, CA, 1985.
8. Denecker, M. and De Schreye, D., On the Duality of Abduction and Model Generation, in: *Proceedings of the International Conference on FGCS*, Tokyo, 1992, pp. 650–657.
9. Dung, P. M., Negation as Hypothesis: An Abductive Foundation for Logic Programming, in: Furukawa, K. (ed.), *Proceedings of 8th International Conference on Logic Programming*, MIT Press, Paris, 1991.
10. Dung, P. M., On the Relations Between Stable and Well-Founded Semantics of Logic Programs, *Theoretical Computer Science* 105:7–25 (1992).
11. Dung, P. M., On the Acceptability of Arguments and Its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and N-Person Game, *Artificial Intelligence* (to appear); an extended abstract appeared in Bajcsy R. (ed.), *Proc. of IJCAI'93*, pp. 852–859.
12. Dung, P. M., An Argumentational Semantics for Logic Programming with Classical Negation, in: Warren, D. S. (ed.), *Proceedings of Tenth International Conference on Logic Programming*, MIT Press, Budapest, 1993.
13. Dung, P. M. and Ruamviboonsuk, P., Well-Founded Reasoning with Classical Negation, in: Nerode, A., Marek, W., and Subramanian, V. S. (eds.), *Proceedings of the First International Workshop on Logic Programming and Nonmonotonic Reasoning*, MIT Press, Washington, 1991.
14. Eshghi, K. and Kowalski, R. A., Abduction Compared with Negation by Failure, in: *Proceedings of 6th International Conference on Logic Programming*, MIT Press, 1989.
15. Eshghi, K. and Kowalski, R. A., Abduction Compared with Negation by Failure, Technical Report, Department of Computing, Imperial College, 1988.

16. Finger, J. J. and Genesereth, M. R., Residue: A Deductive Approach to Design Synthesis, STAN-CS-85-1035, Stanford University.
17. Fages, F., Consistency of Clark's Completion and Existence of Stable Models, *Journal of Methods of Logic in Computer Sciences* (1993).
18. Giordano, L., Martelli, A., and Sapino, M. L., A Semantics for Eshghi and Kowalski's Abductive Procedure, in: Warren D. S. (ed.), *Proceedings of 10th International Conference on Logic Programming*, MIT Press, Budapest, 1993.
19. Van Gelder, A., Ross, K., and Schlipf, J. S., Unfounded Sets and Well-Founded Semantics for General Logic Programs, *Proc. of PODS*, 1988.
20. Gelfond, M. and Lifschitz, V., The Stable Model Semantics for Logic Programs, Kowalski, R. and Bowen, K. (eds.), *Proceedings of the 5th Int. Conf./Symp. on Logic Programming*, MIT Press, 1988.
21. Kakas, T., Kowalski, R., and Toni, F., Abductive Logic Programming, *J. of Logic and Computation* 2(6):719–770 (1992).
22. Kakas, T. and Mancarella, P., Stable Theories for Logic Programs, in: *Proceedings of the International Logic Programming Symposium*, MIT Press, 1991.
23. Kakas, T. and Mancarella, P., Preferred Extensions are Partial Stable Models, *J. of Logic Programming* 14:341–348 (1992).
24. Kakas, T. and Mancarella, P., Generalized Stable Models: A Semantics for Abduction, in: *Proceedings of the European Conference on AI*, 1990.
25. Kakas, T. and Mancarella, P., Constructive Abduction in Logic Programming, Technical Report, 1993.
26. Kowalski, R. A., Logic Programming and Artificial Intelligence, Invited paper at International Joint Conference on AI, Sydney, 1991.
27. Kunen, K., Signed Data Dependencies in Logic Programming, *J. of Logic Programming* 7:231–245 (1989).
28. Lloyd, J. W., *Foundations of Logic Programming*, 2nd edition, Springer-Verlag, 1987.
29. Marek, W., Nerode, A., and Remmel, J., A Theory of Nonmonotonic Rule Systems I, *Annals of Mathematics and Artificial Intelligence* 1:241–273 (1990).
30. Pereira, L. M., Aparicio, J. N., and Alferes, J. J., Adding Closed World Assumption to Well-Founded Semantics, in: *Proceedings of Fifth Generation Computer Systems*, ICOT, Japan, 1992.
31. Przymusinski, T. C., On the Declarative Semantics of Deductive Databases and Logic Programs, in: Minker, J. (ed.), *Foundations of Deductive Databases & Logic Programming*, Morgan Kaufmann, Palo Alto, CA, 1987.
32. Przymusinski, T. C., Extended Stable Semantics for Normal and Disjunctive Logic Programs, in: *Proceedings of 7th International Conference on Logic Programming*, MIT Press, 1990.
33. Przymusinski, T. C., Every Logic Program has a Natural Stratification and an Iterated Least Fixed Point Model, in: *Proceedings of Principles of database Systems*, 1989, pp. 11–21.
34. Poole, D., A Logical Framework for Default Reasoning, *Artificial Intelligence* 36:27–47 (1988).
35. Przymusinski, T. C., Perfect Model Semantics, in: *Proceedings of 5th International Conference on Logic Programming*, MIT Press, 1988.
36. Przymusinski, T. C., Semantics of Disjunctive Programs and Deductive Databases, in: *Proceedings of Conference on Deductive and Object-Oriented Databases*, 1991.

37. Ross, K. A., A Procedural Semantics for Well-Founded Negation in Logic Programs, Technical Report, Department of Computer Science, Stanford University 1988.
38. Sato, T., Completed Logic Programs and their Consistency, *J. of Logic Programming* 9:33–44 (1990).
39. Sedgewick, R., *Algorithms in C*, Addison-Wesley, 1990.
40. Sacca, D. and Zaniolo, C., Partial Models and Three-Valued Models in Logic Programs with Negation, in: Nerode, A., Marek, W., and Subramanian, V. S. (eds.), *Proc. of First International Workshop on Logic Programming and Nonmonotonic Reasoning*, MIT Press, Washington, 1991.
41. Torres, A., Nondeterministic Well-Founded Semantics, to appear in *Annals of Mathematics and AI.*
42. Touretzky D. S., Horty, J. F., and Thomason R. H., A Clash of Intuitions: The Current State of Nonmonotonic Multiple Inheritance Systems, in: *Proceedings of the International Joint Conference on AI*, 1987.