Nelson's Strong Negation, Safe Beliefs and the Answer Set Semantics

Magdalena Ortiz and Mauricio Osorio

Universidad de las Américas, CENTIA
Sta. Catarina Mártir, Cholula, Puebla
72820 México
{is103378,josorio}@mail.udlap.mx

Abstract In this paper we consider an extension of the answer set semantics allowing arbitrary use of strong negation. We prove that the strong negation extension of any intermediate logic provides a suitable basis for reasoning under the answer set semantics. We propose two new notions of equivalence that are more general than strong equivalence: *substitution equivalence* and *contextualized equivalence*.

1 Introduction

For many years the main line of research in the area of Answer Set Programming has seen semantics in the traditional way of logic programming: reductions on logic programs and fixed-point style definitions[1]. The best known example of such definitions is the *Gelfond-Lifschitz reduct*, the original definition of the semantics [2]. The extensions to wider families of programs that followed were also defined as reductions *à la Gelfond-Lifschitz* : from the introduction of strong negation [3] to nested programs [7], a rather wide range of such reducts has been proposed.

Alternative approaches have been considered like proof theoretic characterizations [9] or inference in different logics [6,8]. However, in contrast to reductions, they are often seen more as theoretical tools than as definitions of the semantics. In this paper we consider one of these alternative approaches: logic programs can be understood as propositional theories and their answer sets are then defined as models in a formal logic system. In particular, we follow the line of research started by Pearce [17], who focused on establishing links between negation in the stable model semantics and negation in logic [16]. As Pearce points out, the standard *default negation* in stable inference can be characterized by negation in Heyting's *intuitionistic logic*. Using the logic **HT** (an extension of intuitionistic logic, see Section 2.1), he defined *equilibrium logic*, which became very well accepted in the context of purely logical approaches to the answer set semantics. A similar formalism called *safe beliefs* was introduced by Osorio et al. [13,14] establishing new results on the correspondence between answer sets and super-intuitionistic logics. The answer set semantics of disjunctive and nested programs can be seen as particular instances of both equilibrium models and safe beliefs. Additionally

[1] The terms *answer set semantics* and *stable model semantics* are considered synonyms for the purposes of this work.

they define an extension of answer sets for theories with even more flexible syntax, for example for programs that contain the implication connective in the body of rules [15].

1.1 About this Work

Since it was introduced in [3], strong negation has been well accepted in the answer set programming community[2]. However, this connective has not received a fair treatment. While the answer set semantics has been extended to always more flexible classes of logic programs where conjunctions, disjunctions and default negations are allowed to occur unrestrictedly in any part of the formulas, strong negation has remained tied to the atomic level. In some cases, to compute the semantics, strong negation is removed from the program and 'simulated' introducing new atoms and constraints. Even the purely logical approaches, less syntactically restricted, have often focused on programs that do not contain strong negation [11,12]. Nevertheless, this connective is important from the knowledge representation and application development perspective. Also from a theoretical point of view, understanding its behaviour brings interesting insights about constructive negation and negation in logic programming. We will study this connective in more detail here. In particular we analyse the repercussion of its unrestricted use in the answer set semantics.

It was also Pearce who provided core insights on the issue of strong negation. Based on Nelson's extension of intuitionistic logic with a new negation connective, Pearce enhanced equilibrium logic with the same connective. He proved that this extended logic characterizes the answer sets semantics of programs with strong negation [16]. Despite this correspondence, which lets the answer sets community take advantage of existing work in Nelson's logics, the issue had not been addressed in detail. Many questions remained open, despite the availability of all machinery required to answer them. We will list only a number of them:

1. What are the effects of allowing the arbitrary use of strong negation in logic programs? Are the existing results about equivalence, transformations, etc. still applicable?
2. Are all Nelson extensions of intermediate logics invariant w.r.t. the answer set semantics?
3. Can strong negation be effectively eliminated from any logic program or formula in an unified way?

We will answer these questions in the current work. More specifically, we will address 2 and 3 in Section 3. Then we will move to answering item 1, which will take us through sections 4 and 5 where we will extend existing results in answer sets to the case of arbitrary theories with two kinds of negation. In particular, we provide notions of equivalence which are more general than strong equivalence. They allow us to do program transformations that can not be properly captured by strong equivalence alone. Some of them are also relevant in the absence of strong negation.

[2] By *strong negation* we mean *classical negation* in [3]. See Section 2.2.

2 Preliminaries

2.1 Intermediate Logics and Strong Negation Extensions

We will represent classical propositional logic by the symbol $\mathbf{C}$ and Heyting's Intuitionistic Logic by $\mathbf{I}$. The set $\mathcal{I} := \{\wedge, \vee, \rightarrow, \bot\}$ denotes the connectives of $\mathbf{I}$. $\mathbf{I}$-formulas are the formulas built from the connectives in $\mathcal{I}$ in the standard way. The symbol $\neg$ will be called *intuitionistic negation* and $\neg\alpha$ is an abbreviation of $\alpha \rightarrow \bot$ for any formula α. We will denote by $\mathcal{N} := \mathcal{I} \cup \{\sim\}$ the set of connectives of Nelson's logics, where $\sim$ is an unary connective called *strong negation*[3]. The formulas constructed using the connectives in $\mathcal{N}$ are called $\mathbf{N}$-formulas.

For any formulas α, β, will use the following abbreviations: $\top := \bot \rightarrow \bot$, $\alpha \leftrightarrow \beta := (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$. For a logic $\mathbf{X}$ and a formula α, the expression $\vdash_{\mathbf{X}} \alpha$ represents the standard derivability relation in logic $\mathbf{X}$, i.e. α can be derived from the axioms in $\mathbf{X}$ using modus ponens as the only inference rule. $\beta \vdash_{\mathbf{X}} \alpha$ can be read as $\vdash_{\mathbf{X}} \beta \rightarrow \alpha$ and $\beta \dashv\vdash_{\mathbf{X}} \alpha$ is an abbreviation of $\beta \vdash_{\mathbf{X}} \alpha$ and $\alpha \vdash_{\mathbf{X}} \beta$.

An *atom* is a propositional variable. A *$\sim$-literal* is either an atom a or its strong negation $\sim a$. When we use the term *literal* alone, we mean a $\sim$-literal l or its weak negation $\neg l$. A theory is a set of formulas. In a slight abuse of notation, we may write any finite theory Γ as a formula γ, where $\gamma := \bigwedge_{\varphi \in \Gamma} \varphi$. For any formula φ, the *signature of* φ is the set of propositional variables that occur in φ, as well as the strong negation of each propositional variable, i.e. a set of $\sim$-literals. We will represent it by $\mathcal{L}_\varphi$. Sometimes, we are only interested in the (positive) atoms of a signature. In this case, we will call it *atomic signature* and denote it by $\mathcal{L}^+_\varphi$. For example, given the formula $\alpha := a$, its signature is $\mathcal{L}_\alpha = \{a, \sim a\}$, and its atomic signature $\mathcal{L}^+_\alpha = \{a\}$. Given a set of literals A, we use the notation $\neg A := \{\neg a \mid a \in A\}$; $A^\wedge := \bigwedge_{a \in A} a$, $\widetilde{A}_\varphi := \{a \mid a \in \mathcal{L}_\varphi \setminus A\}$ and $\overline{A}_\varphi := A \cup \neg\widetilde{A}_\varphi$. We will just use the terms $\widetilde{A}$ and $\overline{A}$ when the formula φ is clear by context.

An axiomatic formalization of $\mathbf{I}$ is given in [10]. By adding additional axioms to intuitionistic logic, we obtain the logics that are usually known as *intermediate* or *super intuitionistic* logics. Here we will use the term $\mathbf{I}$*-logic* to refer to any axiomatic extension of $\mathbf{I}$, that is strictly weaker than $\mathbf{C}$ [4]. $\mathbf{I}$-logics form a lattice in which the supreme is the unique lower cover of $\mathbf{C}$. Many names for this logic can be found in the literature, like *Smetanich logic*, the logic of *Here and There* ($\mathbf{HT}$)[5], etc. We will refer to it as $\mathbf{G_3}$, since it is also the three valued logic used by Gödel. It can be obtained by adding to $\mathbf{I}$ the axiom schema $(\neg q \rightarrow p) \rightarrow (((p \rightarrow q) \rightarrow p) \rightarrow p)$.

[3] Unfortunately, there is a mismatch between the traditional notation in intermediate logics and the one used by the answer set community, where $\neg$ often denotes strong negation. We will adhere to the intermediate logics standard, since we want to emphasize that negation in these logics properly captures negation in answer sets.

[4] In a slight abuse of notation we will use the same symbol to refer to a formal logic system as well as to its theorems. We say that $\mathbf{X}$ is strictly weaker than $\mathbf{Y}$ iff $\mathbf{X} \subsetneq \mathbf{Y}$.

[5] This is the name usually found in he works of Pearce. The name is due to the fact of it being characterized by the Kripke frames containing exactly two worlds: the *here* world and the *there* world.

For any **I**-logic $\mathbf{X}$, we can add the following axiom schemata to the axiomatization of $\mathbf{X}$ to obtain $\mathbf{N}(\mathbf{X})$, the *least strong negation extension* of $\mathbf{X}$:

N1. $\sim(\alpha \to \beta) \leftrightarrow \alpha \wedge \sim\beta$
N2. $\sim(\alpha \wedge \beta) \leftrightarrow \sim\alpha \vee \sim\beta$
N3. $\sim(\alpha \vee \beta) \leftrightarrow \sim\alpha \wedge \sim\beta$
N4. $\alpha \leftrightarrow \sim\sim\alpha$
N5. $\sim\neg\alpha \leftrightarrow \alpha$
N6. $\sim\alpha \to \neg\alpha$ for atomic α

These axioms are introduced in [20,21], where they were added to **I**. Since $\sim\alpha \to \neg\alpha$ is a theorem for any α (not only atoms), the $\sim$ connective is called strong negation. It was introduced by Nelson in [1] and intuitively it means that something is known to be false, not only assumed false due to the absence of a proof. We will use the term **N**-logics to refer to the least strong negation extensions of **I**-logics. In particular, we will denote $\mathbf{N}(\mathbf{I})$ by **N** and $\mathbf{N}(\mathbf{G_3})$ by $\mathbf{N_5}$ [6]. **N**-logics form a lattice that was studied in detail by Kracht [4]. We will denote as $\mathbf{I}(\mathbf{Y})$ the set of **I**-formulas that are theorems of an **N**-logic $\mathbf{Y}$ and call it the **I**-*fragment* of $\mathbf{Y}$. Since the axiom schemata **N1**. to **N6**. do not add any theorems in the basic language of an **I**-logic, $\mathbf{I}(\mathbf{N}(\mathbf{X}))$ = $\mathbf{X}$ is an **I**-logic.

We will recall some of the results from [4] that we will use in the following sections.

Definition 1 (Standard Form). *An* **N***-formula is said to be in* standard form *if its built from* $\sim$*-literals using only intuitionistic connectives.*

A formula in standard form has all occurrences of the $\sim$ connective just in front of an atom. It is easy to see that for any given formula, due to axioms **N1** to **N6**, the $\sim$ connective can be pushed in until the formula is in standard form. We will denote by $s(\varphi)$ the standard form of an **N**-formula φ. In the following definition we present a transformation of **N**-formulas into **I**-formulas.

Definition 2. *Let* φ *be any* **N***-formula in standard form and let* $\mathcal{L}'$ *be a signature of the same cardinality as* $\mathcal{L}^+_\varphi$ *such that* $\mathcal{L}' \cap \mathcal{L}^+_\varphi = \emptyset$*. We define a* variable twisting function *as a bijective function* $t : \mathcal{L}^+_\varphi \longrightarrow \mathcal{L}'$*. For any* $p \in \mathcal{L}\varphi^+$ *the atom* $t(p)$ *will be called the* twistor of p *and denoted by* p'*. The mapping of* φ *to the* **I***-formula* φ° *over the signature* $\mathcal{L}^+_\varphi \cup \mathcal{L}'$ *is defined recursively as follows:*

1. *If* $\varphi = \bot$*, then* $\varphi^\circ := \bot$
2. *If* $\varphi = a$*, where* a *is any atom, then* $\varphi^\circ := a$
3. *If* $\varphi = \sim a$*, where* a *is any atom, then* $\varphi^\circ := t(a) = a'$
4. *If* $\varphi = \alpha \# \beta$*, where* α *and* β *are any pair of formulas and* $\# \in \{\wedge, \vee, \to\}$*, then* $\varphi^\circ := \alpha^\circ \# \beta^\circ$

The constraint set *of* φ*, written as* Ψ_φ *is defined as follows:* $\Psi_\varphi := \bigwedge_{p \in \mathcal{L}^+_\varphi} p' \to \neg p$*.* Ψ_Γ *is just an abbreviation of* $\bigwedge_{\gamma \in \Gamma} \Psi_\gamma$*.*
Finally, define $\varphi^\oplus := \Psi_\varphi \wedge \varphi^\circ$ *and* $\varphi_\bowtie := \Psi_\varphi \to \varphi^\circ$*. The formula* $\varphi_\bowtie$ *will be called* twist of φ*.*
For any **N***-formula* ψ *that is not in standard form, take* $\psi_\bowtie := s(\psi)_\bowtie$ *and* $\psi^\oplus := s(\psi)^\oplus$*.*

[6] The name is due to its semantic characterization in terms of 5-valued truth tables. See [18].

The following propositions show the relevance of the twist of a formula [7]:

Proposition 1 (Theorem 8 in [4]). *For any* **I***-logic* $\mathbf{X}$ *and* **N***-formula* φ, $\vdash_{\mathbf{N}(\mathbf{X})} \varphi$ *iff* $\vdash_{\mathbf{X}} \varphi_{\bowtie}$.

Proposition 2 (Theorem 9 in [4]). *For any* **N***-logic* $\mathbf{X}$ *and* **I***-formula* φ, $\vdash_{\mathbf{X}} \varphi$ *iff* $\vdash_{\mathbf{I}(\mathbf{X})} \varphi$.

As for the second rewriting given in the previous definition, $\varphi^{\oplus}$, its importance will be made clear when we state Theorem 1 in Section 3.

2.2 General Notation

A formula α is defined to be $\sim$-inconsistent in logic $\mathbf{X}$ if both $\alpha \vdash_{\mathbf{X}} \beta$ and $\alpha \vdash_{\mathbf{X}} \sim\beta$ hold for some β. A formula α is defined to be $\neg$-inconsistent in logic $\mathbf{X}$ if both $\alpha \vdash_{\mathbf{X}} \beta$ and $\alpha \vdash_{\mathbf{X}} \neg\beta$ hold for some β. Since it is easy to see that $\sim$-inconsistency and $\neg$-inconsistency imply each other, we will just refer to formulas that are inconsistent in logic $\mathbf{X}$. For any **I**-logics (**N**-logics) $\mathbf{X}$ and $\mathbf{Y}$, if a formula α is inconsistent in $\mathbf{X}$ then α is inconsistent in $\mathbf{Y}$, hence we may omit the logic and say that α is inconsistent. A formula α is defined to be *complete in logic* $\mathbf{X}$ *w.r.t. a signature* $\mathcal{L}$ if for any formula β such that $\mathcal{L}_\beta \subseteq \mathcal{L}$, either $\alpha \vdash_{\mathbf{X}} \beta$ or $\alpha \vdash_{\mathbf{X}} \neg\beta$.

Logic programs as propositional formulas. A logic program is usually defined as a set of rules. In our approach, logic programs are simply particular cases of propositional theories with a restricted syntax. For the purpose of this work the terms formula, program and theory might be considered equivalent. We will use the standard logic notation to refer to logic programs. A rule that would be written as $H \leftarrow B$ or $H :- B$ in logic programming is represented by the formula $B \rightarrow H$. Conjunctions and disjunctions are denoted by the $\wedge$ and $\vee$ connectives. In order to be consistent with the notation used for **N**-logics, the *negation as failure* connective denoted not in logic programs is here represented by the symbol $\neg$, and *strong negation* is denoted with the symbol $\sim$. We will use the name **N**-nested formula for formulas of the form $\alpha \rightarrow \beta$ where both α and β are **N**-formulas in standard form with no occurrences of the $\rightarrow$ connective. An **I**-nested formula is an **N**-nested formula with no occurrences of $\sim$.

2.3 $\mathbb{I}$-Safe Beliefs, Equilibrium Models and Answer Sets

$\mathbb{I}$-Safe Beliefs. We will briefly recall *safe beliefs*, the work of Osorio et.al. [13,14], in order to extend their results to propositional theories with strong negation.

Definition 3 (X-Safe Beliefs). *Let* $\mathbf{X}$ *be any* **I***-logic and let* φ *be any* **I***-formula. A set* M *of atoms such that* $\varphi \wedge (\neg\neg\overline{M})^{\wedge} \dashv\vdash_{\mathbf{X}} \overline{M}^{\wedge}$ *is called a* $\mathbf{X}$*-safe belief of* φ.

[7] Since in [4] **N**-logics are given an algebraic treatment, both theorems are stated for varieties of algebras. Here we present them in terms of theoremhood.

Note that we simply write $\varphi \wedge (\neg\neg\overline{M})^{\wedge} \dashv\vdash_{\mathbf{X}} \overline{M}^{\wedge}$ instead of $\varphi \wedge (\neg\neg\overline{M}_{\varphi})^{\wedge} \dashv\vdash_{\mathbf{X}} \overline{M}_{\varphi}^{\wedge}$. When we talk about safe beliefs, we will usually do this simplification on the notation [8].

Intuitively, we can explain safe beliefs as follows: Our formula or program represents our knowledge. We want its semantics to be a complete theory. For some atoms a in the language, either a or $\neg a$ can be infered directly from the information in the program. If this is not the case, we will have to complete our theory by making some additional assumptions. However, we want to be cautious and assume as little information as possible. Hence we only assume a set of weakly negated literals. From an intuitionistic perspective, this can be read as assuming that we have no proof of the truth or falsity of the assumed literals, which seems cautious enough. If this weak assumptions are enough to complete our theory without falling into inconsistency, then we have a safe belief.

In Theorem 4.1 in [14] the authors prove that $\mathbf{X}$-safe beliefs are equivalent for any $\mathbf{I}$-logic $\mathbf{X}$. Hence we may refer to safe beliefs without mentioning a particular $\mathbf{I}$-logic. We will call $\mathbb{I}$-safe belief of φ a set of atoms M such that M is a $\mathbf{X}$-safe belief of φ for any $\mathbf{I}$-logic $\mathbf{X}$.

Equilibrium Models. The original definition of an equilibrium model of an $\mathbf{I}$-formula can be found in [17], where they are defined as $\mathbf{G_3}$ models satisfying some particular conditions. Since the equivalence of $\mathbb{I}$-safe beliefs and equilibrium models has been proved (see Proposition 3.1 in [14] and Proposition 2.4 in [17]), we might consider equilibrium models and $\mathbb{I}$-safe beliefs as synonyms. For the case of $\mathbf{N}$-formulas, Pearce extended his characterization of equilibrium models using $\mathbf{N_5}$ models [16]. The safe beliefs counterpart of this extension will be given in Section 3.

Answer Sets. We will not present the traditional definition of the answer set semantics. As we have mentioned, it is defined through the Gelfond-Lifschitz reduct [2]. This reduction has been extended to families of programs with less restricted syntax. The most general reduct proposed until now that is relevant for the current work is the one for $\mathbf{N}$-nested formulas given in [7]. However, for other classes of formulas (for example, formulas containing embedded implications) there is no obvious way to extend the reduction.

Both equilibrium inference and $\mathbb{I}$-safe beliefs were proposed as characterizations of the answer set semantics. The equivalence of both formalisms and answer sets for $\mathbf{I}$-nested formulas was proved independently. Lemma 3 in [5] states that a set of atoms is an equilibrium model of an $\mathbf{I}$-nested formula iff it is an answer set of it. Equivalently, Corollary 4.1 in [14] stablishes the same result for safe beliefs. Additionally, these formalisms give a semantics to all $\mathbf{I}$-formulas and are a natural generalization of answer sets for the cases where extended reducts are not available. In this work, we will consider them as the definition of the answer sets semantics for arbitrary $\mathbf{I}$-formulas.

[8] We are using the $\dashv\vdash$ symbol, instead of $\vdash$ used in [13]. The equivalence of both notations in the context of this definition follows from completeness of $\overline{M}^{\wedge}$.

3 $\mathbb{N}$-Safe Beliefs

Now that we have introduced the necessary preliminaries, we will provide an equivalent of **I**-safe beliefs for **N**-formulas. We do it in the natural way:

Definition 4 (X-Safe Beliefs). *Let* $\mathbf{X}$ *be any* $\mathbf{N}$*-logic. Let* φ *be an N-formula and* M *a set of* $\sim$*-literals.* M *is called a* $\mathbf{X}$*-safe belief of* φ *if* $\varphi \wedge (\neg\neg\overline{M})^{\wedge} \dashv\vdash_{\mathbf{X}} \overline{M}^{\wedge}$ *holds.*

Note that as we used sets of atoms when talking about **I**-Safe Beliefs, now we will refer to sets of $\sim$-literals. The reader should keep in mind that we do not exclude inconsistent sets of $\sim$-literals (i.e. sets that contain both a and $\sim a$ for some atom a). Due to this assumption, an inconsistent formula does not have any safe beliefs. It is a common practice in the answer sets community to consider only consistent sets of $\sim$-literals as potential answer sets. In the case of inconsistent formulas, the entire signature is by definition the only answer set. **X**-safe beliefs generalize answer sets modulo this consideration. The definition of **X**-safe beliefs can be restricted to consistent sets of $\sim$-literals in the natural way.

To be able to relate safe beliefs of an **N**-formula φ with **I**-safe beliefs and answer sets, we will use the $^{\oplus}$ function in Definition 2. In the context of the answer set semantics, strong negation is usually eliminated from logic programs by a simple transformation, initially proposed by Gelfond and Lifschitz [3]. This transformation coincides with $\varphi^{\oplus}$ for the restricted case where φ is a disjunctive formula. The authors proved that a consistent set of $\sim$-literals M is an answer set of φ iff M° is an answer set of $\varphi^{\oplus}$. The $^{\oplus}$ transformation became the standard way to approach logic programs with strong negation. In [5] it was extended to nested logic programs. The definition of $\varphi^{\oplus}$ for an arbitrary **N**-formula φ provides a natural generalization of this result. Now we prove that under this transformation, **X**-safe beliefs for any **N**-logic **X** coincide with $\mathbb{I}$-safe beliefs. This is the first of our key results, since it provides a uniform way to eliminate strong negation from arbitrary **N**-formulas under the safe beliefs semantics.

Theorem 1. *Let* $\mathbf{X}$ *be any* **I***-logic,* M *a set of* $\sim$*-literals and* φ *an* **N***-formula.* M *is an* $\mathbf{N}(\mathbf{X})$*-Safe belief of* φ *iff* M° *is a* $\mathbb{I}$*-Safe belief of* $\varphi^{\oplus}$*.*

Proof (Sketch). With Definition 2 as well as Propositions 1 and 2, the reader can easily verify that $\varphi \wedge (\neg\neg\overline{M})^{\wedge} \vdash_{\mathbf{N}(\mathbf{X})} \overline{M}^{\wedge}$ iff $\varphi^{\oplus} \wedge ((\neg\neg\overline{M})^{\wedge})^{\circ} \vdash_{\mathbf{X}} \overline{M^{\circ}}^{\wedge}$. That $\overline{M}^{\wedge} \vdash_{\mathbf{N}(\mathbf{X})} \varphi \wedge (\neg\neg\overline{M})^{\wedge}$ iff $\overline{M^{\circ}}^{\wedge} \vdash_{\mathbf{X}} \varphi^{\oplus} \wedge ((\neg\neg\overline{M})^{\wedge})^{\circ}$ follows from completeness of $\overline{M}^{\wedge}$ (resp. $\overline{M^{\circ}}^{\wedge}$) w.r.t. $\mathcal{L}_{\varphi}$ (resp. $\mathcal{L}_{\varphi^{\oplus}}$).

Our second important result is to answer a question that remained open until now. A **N**-formula has exactly the same **X**-safe beliefs in every **N**-logic **X**.

Theorem 2. *Let* φ *be an* **N***-formula and let* $\mathbf{X}$*,* $\mathbf{Y}$ *be any two* **N***-logics. A set of literals* M *is a* $\mathbf{X}$*-Safe Belief of* φ *iff* M *is a* $\mathbf{Y}$*-safe belief of* φ*.*

Proof (Sketch). Theorem 4.1 in [14] proves that **I**-safe beliefs are invariant w.r.t. the $\mathbb{I}$-safe beliefs semantics. From this result and Theorem 1, it follows that a set of $\sim$-literals M is a $\mathbf{N_5}$-safe belief of an **N**-formula φ iff M is a **N**- safe belief of φ, hence the result also holds for any **N**-logic stronger than **N** and weaker than $\mathbf{N_5}$.

We have proved that **N**-logics are invariant w.r.t. the safe beliefs semantics, hence we may omit particular logics when refering to **N**-safe beliefs. A set M of $\sim$-literals such that M is a **X**-safe belief of γ for every **N**-logic **X** will be called a $\mathbb{N}$-safe belief of γ. Now we can use $\mathbb{N}$-safe beliefs to correctly characterize the answer set semantics of arbitrary **N**-formulas. It is worth pointing out that these results are not really surprising. The relation between Pearce's equilibrium models and $\mathbb{N}$-safe beliefs is clear. In particular, $\mathbf{N_5}$-safe beliefs are just a notational variation of equilibrium models (with strong negation). Already in [17] Pearce had proved that equilibrium models are a generalization of the answer set semantics of nested programs. In the rest of this work, answer sets of an arbitrary **N**-formula φ will be, by definition, the $\mathbb{N}$-safe beliefs of φ.

4 Substitution in N-logics

One peculiar feature of **N**-logics is that the standard substitution theorem does not hold in general and some restricted versions of it have to be used instead. We devote this section to substitution, since it will play a key role in the rest of our work.

First, we will consider *standard* substitution, here represented with the usual notation: $\alpha[\beta/p]$ will denote the formula that results from substituting the formula β for the atom p, wherever the atom occurs in the formula.

Intuitively, in a **N**-formula in standard form, each $\sim$-literal represents a different atom. Hence, sometimes we would like to apply a substitution replacing either all positive occurrences of a particular atom, or all its negative instances. For practical purposes, we will introduce a special notation for his kind of substitution. Given a formula φ in standard form, and an $\sim$-literal l, $\varphi[\alpha \parallel l]$ will denote the formula that results from substituting the occurrences of the $\sim$-literal l by the formula α, but leaving all occurrences of the complementary literal unchanged. If l is of the form a for some atomic a, the occurrences of $\sim a$ remained untouched. Analogously, if l is $\sim a$, strongly negated occurrences of a are substituted.

4.1 Cautious Substitution

Additionally, we will introduce another type of substitution, which will be called *cautious substitution* and denoted as $\alpha\{\beta/p\}$ for formulas α, β and an atom p. It can also be denoted $\alpha\{^0\beta/p\}$ and its recursive definition is as follows:

$$\alpha\{^0\beta/p\} := \begin{cases} \beta & \text{if } \alpha \text{ is the atom } p \\ \alpha & \text{if } \alpha \text{ is } \bot \text{ or an atom different from } p \\ \alpha_1\{^0\beta/p\}\#\alpha_2\{^0\beta/p\} & \text{if } \alpha \text{ is a formula of the form } \alpha_1\#\alpha_2 \\ & \text{where } \# \text{ is either } \wedge \text{ or } \vee \\ \alpha_1\{^0\beta/p\} \rightarrow \alpha_2\{^0\beta/p\} & \text{if } \alpha \text{ is a formula of the form } \alpha_1 \rightarrow \alpha_2 \\ \sim\alpha_1\{^1\beta/p\} & \text{if } \alpha \text{ is a formula of the form } \sim\alpha_1 \end{cases}$$

$$\alpha\{^1\beta/p\} := \begin{cases} \alpha & \text{if } \alpha \text{ is } \bot \text{ or an atom} \\ \alpha_1\{^1\beta/p\}\#\alpha_2\{^1\beta/p\} & \text{if } \alpha \text{ is a formula of the form } \alpha_1\#\alpha_2 \\ & \text{where } \# \text{ is either } \wedge \text{ or } \vee \\ \alpha_1\{^0\beta/p\} \rightarrow \alpha_2\{^1\beta/p\} & \text{if } \alpha \text{ is a formula of the form } \alpha_1 \rightarrow \alpha_2 \\ \sim\alpha_1\{^0\beta/p\} & \text{if } \alpha \text{ is a formula of the form } \sim\alpha_1 \end{cases}$$

Intuitively, the key issue of our *cautious substitution* is that it takes into account the scope of strong negation. If we apply it to a formula in standard form, it coincides with standard substitution (considering each $\sim$-literal as a different atom). For formulas that are not in standard form, the following property assures a desirable behavior of cautious substitution:

Proposition 3. *Let ψ, α be any* **N***-formulas and p an atom. Then $s(\psi\{\alpha/p\}) = s(s(\psi)[\alpha \parallel p])$.*

Proof (Sketch). By straightforward induction on the formula prove that $s(\psi\{^0\alpha/p\}) = s(s(\psi)[\alpha \parallel p])$ and $s(\sim(\psi\{^1\alpha/p\})) = s(s(\sim\psi)[\alpha \parallel p])$.

4.2 Substitution theorems for N-logics

A particular feature of **N**-logics is that the symbol $\leftrightarrow$ does not define a congruential relation on formulas: it can be the case that $\vdash_{\mathbf{X}} \alpha \leftrightarrow \beta$ holds, but $\vdash_{\mathbf{X}} \sim\alpha \leftrightarrow \sim\beta$ doesn't. Thus, when we refer to *equivalence* of formulas, we will have to be more precise and make some particular considerations. The term *weak equivalence* will mean $\vdash_{\mathbf{X}} \alpha \leftrightarrow \beta$. We will use the abbreviation $\vdash_{\mathbf{X}} \alpha \Leftrightarrow \beta$ when both $\vdash_{\mathbf{X}} \alpha \leftrightarrow \beta$ and $\vdash_{\mathbf{X}} \sim\alpha \leftrightarrow \sim\beta$ hold. This stronger condition will be called **N**-*equivalence* and in contrast to weak equivalence, defines a congruential relation on **N**-formulas.

After these remarks we can present the substitution theorem for **N**-logics. The proof of it is available in [19].

Theorem 3 (Substitution theorem for N-logics). *Let α, β and ψ be* **N***-formulas and let p be an atom. Let* **X** *be any* **N***-logic. If $\vdash_{\mathbf{X}} \alpha \Leftrightarrow \beta$ then $\vdash_{\mathbf{X}} \psi[\alpha/p] \leftrightarrow \psi[\beta/p]$.*

As we can see, to be able to substitute we usually require **N**-equivalence of formulas to hold. However, in certain cases this condition may be too strong. We will see two particular cases where weak equivalence suffices. The first is when substitution is not done inside the scope of a $\sim$ symbol. The second is when we use cautious substitution.

Theorem 4. *Let α, β and ψ be* **N***-formulas and let p be an atom such that p does not occur in ψ within the scope of a $\sim$ symbol. Let* **X** *be any* **N***-logic. If $\vdash_{\mathbf{X}} \alpha \leftrightarrow \beta$ then $\vdash_{\mathbf{X}} \psi[\alpha/p] \leftrightarrow \psi[\beta/p]$.*

Proof (Sketch). The proof can be done by a straight forward induction on the construction of ψ. The key point is to observe that when ψ is of the form $\sim\psi'$ then p does not occur in ψ' and hence $\psi[\alpha/p] = \psi[\beta/p] = \psi$.

The following corollary is useful since it states that when formulas are written in standard form, weak equivalence is a sufficient condition to apply substitution. Recall that when we are dealing with $\mathbf{N}$-formulas in standard form, each $\sim$-literal is considered a different atom.

Corollary 1. *Let α, β and ψ be $\mathbf{N}$-formulas such that ψ is in standard form and let l be a $\sim$-literal. Let $\mathbf{X}$ be any $\mathbf{N}$-logic. If $\vdash_{\mathbf{X}} \alpha \leftrightarrow \beta$ then $\vdash_{\mathbf{X}} \psi[\alpha \parallel l] \leftrightarrow \psi[\beta \parallel l]$.*

Proof. Take the formula ψ in Theorem 4 to be ψ°. Take p as a if l is an atom a, and p as a' if l is the strong negation of an atom of the form $\sim a$. Then we have that $\vdash_{\mathbf{X}} \psi^{\circ}[\alpha/p] \leftrightarrow \psi^{\circ}[\beta/p]$. The reader can easily verify that since ψ is in standard form, $\vdash_{\mathbf{X}} \psi^{\circ}[\gamma/p] \leftrightarrow (\psi[\gamma \parallel l])^{\circ}$ for any formula γ. Since $\vdash_{\mathbf{X}} (\psi[\alpha \parallel l])^{\circ} \leftrightarrow (\psi[\beta \parallel l])^{\circ}$ implies $\vdash_{\mathbf{X}} \psi[\alpha \parallel l] \leftrightarrow \psi[\beta \parallel l]$ the corollary holds.

Cautious substitution allows us to establish another form of the substitution theorem for $\mathbf{N}$-logics. This substitution can be safely applied when weak equivalence of formulas holds.

Theorem 5 (Cautious Substitution theorem for $\mathbf{N}$-logics). *Let α, β and ψ be $\mathbf{N}$-formulas and let p be an atom. Let $\mathbf{X}$ be any $\mathbf{N}$-logic. If $\vdash_{\mathbf{X}} \alpha \leftrightarrow \beta$ then $\vdash_{\mathbf{X}} \psi\{\alpha/p\} \leftrightarrow \psi\{\beta/p\}$.*

Proof. Since $\vdash_{\mathbf{X}} \alpha \leftrightarrow \beta$, then by Corollary 1 $\vdash_{\mathbf{X}} s(\psi)[\alpha \parallel p] \leftrightarrow s(\psi)[\beta \parallel p]$. Thus $\vdash_{\mathbf{X}} s(s(\psi)[\alpha \parallel p]) \leftrightarrow s(s(\psi)[\beta \parallel p])$. By Proposition 3, $\vdash_{\mathbf{X}} s(\psi\{\alpha/p\}) \leftrightarrow s(\psi\{\beta/p\})$ and it follows that $\vdash_{\mathbf{X}} \psi\{\alpha/p\} \leftrightarrow \psi\{\beta/p\}$.

5 Equivalence

In logic programming, equivalence is a crucial issue since it allows to replace parts of programs and do transformations on them. Under the answer set semantics two programs Π_1 and Π_2 are said to be *equivalent* when Π_1 has the same answer sets as Π_2. In [5] the authors propose *strong equivalence*, a condition that ensures that equivalence is preserved under extensions of programs. Another very important contribution of [5] is to establish a relationship between equivalence of logic programs and equivalence in logic when logic programs are understood as propositional theories. Following this approach we will propose some new notions of equivalence. As the reader will see, they are natural generalizations of strong equivalence. In this section we will use $\mathcal{SB}(\varphi)$ to denote the set of all sets of $\sim$-literals that are $\mathbb{N}$-safe beliefs of a formula φ.

5.1 Strong, Substitution and Contextualized Equivalence

We will first recall the definition of strong equivalence from [5], rewritten according to our notation. Originally the definition was only stated for the case where α, β and ψ are nested programs, but the authors point out that it can also be read in the context of arbitrary theories as we do here.

Definition 5 (Strong Equivalence). *Let α and β be two $\mathbf{N}$-formulas. α is* strongly equivalent *to β if for any formula ψ, $\mathcal{SB}(\psi \wedge \alpha) = \mathcal{SB}(\psi \wedge \beta)$.*

This is considered the most useful notion or equivalence since given a program P, we may safely replace a set of rules R that are part of P by a new set of rules S as long as R and S are strongly equivalent. However, strong equivalence has a restriction: we can only replace entire rules in a program and not parts of them, i.e. only conjuncts in a formula. Here we face this syntactical restriction and define a new type of equivalence. It generalizes strong equivalence in the following sense: suppose that given a program Π, we are interested in replacing not a set of rules in Π, but just parts of some rules. This can be useful for some program transformations. The equivalence we introduce for this purpose will be called *substitution equivalence*. It is more general than strong equivalence, since formulas that are substitution equivalent are also strongly equivalent, but the converse is not always true.

Definition 6 (Substitution Equivalence). *Let α and β be two* $\mathbf{N}$*-formulas. α is* substitution equivalent *to β if for any formula ψ, $\mathcal{SB}(\psi[\alpha/p]) = \mathcal{SB}(\psi[\beta/p])$ and $\mathcal{SB}(\psi[\sim\alpha/p]) = \mathcal{SB}(\psi[\sim\beta/p])$.*

We are also interested in another kind of equivalence, since in many cases, both strong and substitution equivalence are too strong. Two programs might not be equivalent if we see them independently, but if some particular conditions hold in them, then transformations might be safely applied.

Definition 7 (Contextualized Strong Equivalence). *Let α, β, θ be* $\mathbf{N}$*-formulas. Then α and β are* strongly equivalent in the context of *θ iff for every* $\mathbf{N}$*-formula ψ, $\mathcal{SB}(\theta \wedge \psi \wedge \alpha) = \mathcal{SB}(\theta \wedge \psi \wedge \beta)$.*

Definition 8 (Contextualized Substitution Equivalence). *Let α, β, θ be* $\mathbf{N}$*-formulas. Then α and β are* substitution equivalent in the context of *θ iff for every* $\mathbf{N}$*-formula ψ, $\mathcal{SB}(\theta \wedge \psi[\alpha/p]) = \mathcal{SB}(\theta \wedge \psi[\beta/p])$ and $\mathcal{SB}(\theta \wedge \psi[\sim\alpha/p]) = \mathcal{SB}(\theta \wedge \psi[\sim\beta/p])$.*

As we can see, strong equivalence and substitution equivalence are only special cases of equivalence in the context of $\top$.

5.2 Characterizing Equivalence

Since [5] it has been recognised that using the traditional notions of equivalence in logic to study equivalence of logic programs has remarkable advantages. For formulas that do not contain strong negation, equivalence in logic $\mathbf{G_3}$ characterizes strong equivalence.

Theorem 6 ([5]). *Let α, β be two arbitrary* $\mathbf{I}$*-formulas. α and β are strongly equivalent iff $\vdash_{\mathbf{G_3}} \alpha \leftrightarrow \beta$.*

This result has also been generalized to the case of $\mathbf{N}$-formulas. In [5], the authors state that strong equivalence of $\mathbf{N}$-nested formulas corresponds to equivalence in $\mathbf{N_5}$[9]. Here we present the same result in a slightly more general setting, contextualized strong equivalence of arbitrary $\mathbf{N}$-formulas. However, the generalization is trivial and the results follow straightforwardly from the ones in [5].

[9] In [5] the authors only refer to equivalence in $\mathbf{N_5}$, but they do not say whether they mean $\vdash_{\mathbf{N_5}} \alpha \leftrightarrow \beta$ or $\vdash_{\mathbf{N_5}} \alpha \Leftrightarrow \beta$. By Theorem 4, it is clear that weak equivalence suffices.

Proposition 4. *Let α, β and θ be any* **N***-formulas. Then α is strongly equivalent to β in the context of θ iff $\theta \vdash_{\mathbf{N_5}} \alpha \leftrightarrow \beta$.*

Proof (Sketch). ($\Leftarrow$) Since $\theta \vdash_{\mathbf{N_5}} \alpha \leftrightarrow \beta$ implies $\vdash_{\mathbf{N_5}} (\theta \wedge \alpha) \leftrightarrow (\theta \wedge \beta)$, we know that $\theta \wedge \alpha$ is strongly equivalent to $\theta \wedge \beta$, hence $\mathcal{SB}(\theta \wedge \alpha \wedge \psi) = \mathcal{SB}(\theta \wedge \beta \wedge \psi)$ for any ψ. ($\Rightarrow$) $\mathcal{SB}(\theta \wedge \alpha \wedge \psi) = \mathcal{SB}(\theta \wedge \beta \wedge \psi)$ for any ψ implies that $(\theta \wedge \alpha)$ is strongly equivalent to $(\theta \wedge \beta)$. From the results in [5], we obtain that that $\vdash_{\mathbf{N_5}} (\theta \wedge \alpha) \leftrightarrow (\theta \wedge \beta)$, and thus $\theta \vdash_{\mathbf{N_5}} \alpha \leftrightarrow \beta$.

A further generalization can be done in order to prove that substitution equivalence is characterized by **N**-equivalence.

Proposition 5. *Let α, β and θ be any* **N***-formulas. Then α is substitution equivalent to β in the context of θ iff $\theta \vdash_{\mathbf{N_5}} \alpha \Leftrightarrow \beta$.*

Proof. The ($\Leftarrow$) direction follows trivially from the the substitution theorem (Theo. 3). By $\theta \vdash_{\mathbf{N_5}} \alpha \Leftrightarrow \beta$ both $\vdash_{\mathbf{N_5}} \psi[\alpha/p] \wedge \theta \leftrightarrow \psi[\beta/p] \wedge \theta$ and $\vdash_{\mathbf{N_5}} \psi[\sim\alpha/p] \wedge \theta \leftrightarrow \psi[\sim\beta/p] \wedge \theta$ hold for any ψ, and thus $\mathcal{SB}(\psi[\alpha/p] \wedge \theta) = \mathcal{SB}(\psi[\beta/p] \wedge \theta)$ and $\mathcal{SB}(\psi[\sim\alpha/p] \wedge \theta) = \mathcal{SB}(\psi[\sim\beta/p] \wedge \theta)$ so α is substitution equivalent to β in the context of θ.
The ($\Rightarrow$) direction can be proved by contraposition. If $\theta \nvdash_{\mathbf{N_5}} \alpha \Leftrightarrow \beta$, then it must be the case that either $\nvdash_{\mathbf{N_5}} \alpha \wedge \theta \leftrightarrow \beta \wedge \theta$ or $\nvdash_{\mathbf{N_5}} \sim\alpha \wedge \theta \leftrightarrow \sim\beta \wedge \theta$. Hence either $\alpha \wedge \theta$ is not strongly equivalent to $\beta \wedge \theta$, or $\sim\alpha \wedge \theta$ is not strongly equivalent to $\sim\beta \wedge \theta$. Thus there is some φ s.t. either $\mathcal{SB}(\alpha \wedge \theta \wedge \varphi) \neq \mathcal{SB}(\beta \wedge \theta \wedge \varphi)$ or $\mathcal{SB}(\sim\alpha \wedge \theta \wedge \varphi) \neq \mathcal{SB}(\sim\beta \wedge \theta \wedge \varphi)$. Suppose that α is substitution equivalent to β in the context of θ. Then $\mathcal{SB}(\theta \wedge \psi[\alpha/p]) = \mathcal{SB}(\theta \wedge \psi[\beta/p])$ and $\mathcal{SB}(\theta \wedge \psi[\sim\alpha/p]) = \mathcal{SB}(\theta \wedge \psi[\sim\beta/p])$ holds for any ψ. In particular, take ψ to be $\varphi \wedge p$. Then we get that both $\mathcal{SB}(\theta \wedge \varphi \wedge \alpha) = \mathcal{SB}(\theta \wedge \varphi \wedge \beta)$ and $\mathcal{SB}(\theta \wedge \varphi \wedge \sim\alpha) = \mathcal{SB}(\theta \wedge \varphi \wedge \sim\beta)$ hold and we have a contradiction.

Since **N**-equivalence implies weak equivalence of formulas, from Propositions 4 and 5 we get the following corollaries. They state that substitution equivalence ensures strong equivalence.

Corollary 2. *Let α, β and θ be arbitrary* **N***-formulas. If α and β are substitution equivalent in the context of θ, then α and β are strongly equivalent in the context of θ.*

Corollary 3. *Let α, β and θ be arbitrary* **N***-formulas. If α and β are substitution equivalent in the context of θ, then $\sim\alpha$ and $\sim\beta$ are strongly equivalent in the context of θ.*

The converse, however, is not always true. To show that in the general case strong equivalence does not imply substitution equivalence, we provide a counterexample.

Example 1. Let $\alpha := \sim\neg a$ and $\beta := a$. It is easy to verify that $\vdash_{\mathbf{N_5}} \alpha \leftrightarrow \beta$, so we know that α is strongly equivalent to β. Now take the formula $\theta := \sim p \rightarrow b$. We see that $\{a\} \in \mathcal{SB}(\theta[\alpha/p])$, while $\mathcal{SB}(\theta[\beta/p]) = \emptyset$. This proves that α is not substitution equivalent to β. Note that $\nvdash_{\mathbf{N_5}} \sim\alpha \leftrightarrow \sim\beta$.

However, in many cases strong equivalence does imply substitution equivalence, and hence weak equivalence of formulas suffices to arbitrarily replace any part of a formula by another. This is of course the case of **I**-formulas, since weak equivalence collapses with **N**-equivalence.

Corollary 4. *Let α, β and θ be* **I***-formulas. It holds that α and β are substitution equivalent in the context of θ iff α and β are strongly equivalent in the context of θ.*

In general, for **N**-formulas, strong equivalence will suffice to ensure substitution equivalence as long as the proper type of substitution is used. For example, if we consider **N**-formulas in standard form, substitution must be done considering each $\sim$-literal as a different atom. The following corollary is a straightforward consequence of Proposition 4 and Corollary 1.

Corollary 5. *Let α and β be* **N***-formulas. If α and β are strongly equivalent then for any* **N***-formula ψ in standard form, $\mathcal{SB}(\psi[\alpha \parallel l]) = \mathcal{SB}(\psi[\beta \parallel l])$.*

When we consider arbitrary **N**-formulas, strong equivalence implies cautions substitution equivalence. From Proposition 4 and Theorem 5, we get the following corollary.

Corollary 6. *Let α and β be* **N***-formulas. If α and β are strongly equivalent then for any* **N***-formula ψ, $\mathcal{SB}(\psi\{\alpha/p\}) = \mathcal{SB}(\psi\{\beta/p\})$.*

The notions of equivalence we have introduced give us more flexibility than traditional strong equivalence in two senses: first we might do partial transformations on rules, and second we can safely transform logic programs when a certain context makes them equivalent, without requiring equivalence to hold as a stand alone condition. Despite the theoretical nature of the results given in this section, these generalized notions of equivalence have proved to be very useful when providing new results and extending existing ones in the context of answer sets. Some previous proofs have been rewritten in a more compact and general way, and some interesting new results have been achieved. Unfortunately, due to space limitations, they can not be included in the current work.

6 Conclusions

In this paper, we have proposed $\mathbb{N}$-safe beliefs and analyzed the answer set semantics of theories where two types of negation are arbitrarily used. The results concerning elimination of strong negation show that $\mathbb{N}$-safe beliefs are no more expressive than $\mathbb{I}$-safe beliefs. However, we believe that the extension of the answer set semantics to arbitrary propositional formulas with two different negation connectives can be valuable for problem solving and more natural knowledge representation, as well as for technical purposes like program transformations. We claim that when the semantics is bound to a less restricted syntax, writing formal statements is more intuitive and accurate. Most of the results in this work were obtained through the use of strong negation extensions of intermediate logics. Our main goal in this sense is to help providing a clearer understanding of the advantage of logic-based approaches to the answer set semantics. They are not harder to understand than other approaches. Additionally, they allow suitable extensions and provide good intuitions about the semantics in general.

References

1. D. Nelson. Constructible Falsity. *Journal of Symbolic Logic*, 14:16–26, 1949.
2. Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In R. Kowalski and K. Bowen, editors, *5th Conference on Logic Programming*, pages 1070–1080. MIT Press, 1988.
3. Michael Gelfond and Vladimir Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9:365–385, 1991.
4. Marcus Kracht. On extensions of intermediate logics by strong negation. *Journal of Philosophical Logic*, 27(1):49–73, 1998.
5. Vladimir Lifschitz, David Pearce, and Agustin Valverde. Strongly equivalent logic programs. *ACM Transactions on Computational Logic*, 2:526–541, 2001.
6. Vladimir Lifschitz and G. Schwarz. Extended logic programs as autoepistemic theories. In L. M. Pereira and A. Nerode, editors, *2nd International Workshop on Logic Programming & Non-Monotonic Reasoning*, pages 101–114, Cambridge, Mass., 1993. MIT Press.
7. Vladimir Lifschitz, L. R. Tang, and H. Turner. Nested expressions in logic programs. *Annals of Mathematics and Artificial Intelligence*, 25:369–389, 1999.
8. V. W. Marek and M. Truszczyński. Reflexive autoepistemic logic and logic programming. In L. M. and A. Nerode, editors, *2nd International Workshop on Logic Programming & Non-Monotonic Reasoning*, pages 115–131, Cambridge, Mass., 1993. MIT Press.
9. V. Wiktor Marek, Anil Nerode, and Jeffrey B. Remmel. A theory of nonmonotonic rule systems. In *Logic in Computer Science*, pages 79–94, 1990.
10. Elliott Mendelson. *Introduction to mathematical logic (3rd ed.)*. Wadsworth Publ. Co., 1987.
11. Juan Antonio Navarro. Answer set programming through G_3 logic. In Malvina Nissim, editor, *Seventh ESSLLI Student Session, European Summer School in Logic, Language and Information*, Trento, Italy, August 2002.
12. Mauricio Osorio, Juan Antonio Navarro, and José Arrazola. A logical approach for A-Prolog. In Ruy de Queiroz, Luiz Carlos Pereira, and Edward Hermann Haeusler, editors, *9th Workshop on Logic, Language, Information and Computation (WoLLIC)*, volume 67 of *Electronic Notes in Theoretical Computer Science*, pages 265–275, Rio de Janeiro, Brazil, 2002. Elsevier Science Publishers.
13. Mauricio Osorio, Juan Antonio Navarro, and José Arrazola. Applications of intuitionistic logic in answer set programming. *Theory and Practice of Logic Programming*, 4:325–354, 2004.
14. Mauricio Osorio, Juan Antonio Navarro, and José Arrazola. Safe beliefs for propositional theories. Accepted to appear at Annals of Pure and Applied Logic, 2004.
15. Mauricio Osorio and Magdalena Ortiz. Embedded Implications and Minimality in ASP. In Michael Hanus Ulrich Geske Dietmar Seipel and Oskar Bartenstein, editors, *15th International Conference on Applications of Declarative Programming and Knowledge Management. INAP 2004*, Postdam, Germany, March 2004.
16. David Pearce. From here to there: Stable negation in logic programming. In D. M. Gabbay and H. Wansing, editors, *What Is Negation?*, pages 161–181. Kluwer Academic Publishers, Netherlands, 1999.
17. David Pearce. Stable inference as intuitionistic validity. *Logic Programming*, 38:79–91, 1999.
18. David Pearce, Inman P. de Guzmán, and Agustín Valverde. A tableau calculus for equilibrium entailment. In *Proceedings of the International Conference on Automated Reasoning with Analytic Tableaux and Related Methods*, pages 352–367. Springer-Verlag, 2000.
19. Helena Rasiowa. *An Algebraic Approach to Non-Classical Logics*, volume 78 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, Amsterdam, 1974.

20. Nikolay Vorob'ev. Constructive propositional calculus with strong negation. *Doklady Akademii Nauk SSSR*, 85:465–468, 1952. In Russian.
21. Nikolay Vorob'ev. The problem of deducibility in constructive propositional calculus with strong negation. *Doklady Akademii Nauk SSSR*, 85:689–692, 1952. In Russian.