G'$_3$-Stable Semantics and Inconsistency
Semantica G3'-Estable e Inconsistencia

Mauricio Osorio[1], **Claudia Zepeda**[2], **Juan Carlos Nieves**[3] **and José Luis Carballido**[2]
[1]Universidad de las Américas – Puebla, CENTIA
Email: osoriomauri@gmail.com
[2]Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación
Email: {czepedac, jlcarballido7}@gmail.com
[3]Universitat Politècnica de Catalunya
Software Department (LSI)
Email: jcnieves@lsi.upc.edu

Article received on July 19, 2008; accepted on April 03, 2009

Abstract
We present an overview on how to perform non-monotonic reasoning based on paraconsistent logics. In particular, we show that one can define a logic programming semantics based on the paraconsistent logic G'$_3$ which is called G'$_3$-stable semantics. This semantics defines a frame for performing non-monotonic reasoning in domains which are pervaded with vagueness and inconsistencies. In fact, we show that, by considering also a possibilistic logic point of view, one can use this extended framework for defining a possibilistic logic programming approach able to deal with reasoning, which is at the same time non-monotonic and uncertain.
Keywords: G'$_3$-stable semantics, Logic Programming, Non-Monotonic Reasoning.

Resumen
Presentamos un resumen acerca de cómo realizar razonamiento no-monótono basado en lógicas paraconsistentes. En particular, mostramos que es posible definir una semántica de programación lógica basada en la lógica paraconsistente G'$_3$, la cual es llamada semántica G'$_3$-estable. Esta semántica define un marco para realizar razonamiento no-monótono en dominios los cuales están plagados de vaguedades e inconsistencias. De hecho, mostramos que al considerar también un punto de vista lógico posibilista, es posible usar la extensión de este marco de trabajo para definir un enfoque de programación lógica posibilístico que puede tratar con razonamiento que es al mismo tiempo no monótono e incierto.
Palabras Clave: Semántica **:** G'$_3$-estable, Programación lógica, Razonamiento No-Monótono.

1 Introduction

Most logics (at least classical logic and all the constructive intermediate logics) share the property: *Assuming a,* $\neg$ *a one can conclude b, meaning that in the presence of inconsistent* information *one can prove anything (such as the unrelated formula b in this example). Paraconsistent logics* reject this principle; they are not *explosive* in the presence of inconsistent information. Recall that Minsky's "*Frame paper*" (1975) in its original form had an appendix entitled "*Criticism of the Logistic approach*" which states: *"But I do not believe that consistency is necessary or even desirable in a developing intelligent system. No one is ever completely consistent. What is important is how one handles paradox or conflict, how one learns from mistakes, how one turns aside from suspected inconsistencies"* (Minsky, 1975).

We think that paraconsistent logic could help to give an answer to this important issue addressed by Minsky. In fact, in (Osorio et al, 2008) an interesting approach for Knowledge Representation (KR) was proposed. This approach can be supported by any paraconsistent logic stronger than or equal to C_ω, the weakest paraconsistent logic introduced by da Costa.

A second criticism that Minsky gave to the logistic approach is that logic is monotonic: "*Because Logicians are not concerned with systems that will later be enlarged, they can design axioms that permit only the conclusions they*

want. In the development of Intelligence the situation is different. One has to learn which features of situations are important, and which kinds of deductions are not to be regarded seriously" (Minsky, 1975).

Non-monotonic logics were developed as an attempt to solve this problem. Actually the research community has long recognized the study of Non-Monotonic Reasoning (NMR) as a promising approach to model features of commonsense reasoning. On the other hand, monotonic logics have been successfully applied as a basic building block in the formalization of non-monotonic reasoning. The original idea, suggested by McDermott and Doyle (McDermott and Doyle, 1980), was to use well known modal logics. McDermott, in (McDermott, 1982), attempted to define non-monotonic logics based on the standard T, S4 and S5 logics. But he observed that, unfortunately, the non-monotonic version of S5 collapses to the ordinary logic S5. *Grounded* non-monotonic logic is a proposed solution to this problem. The term *grounded* in non-monotonic logics refers to the idea of enabling the agent to make only assumptions that are "*grounded*" in the world's knowledge. According to (Donini et al, 1997), the notion of groundedness was actually introduced by Konolige (Konolige, 1987). It is worth mentioning that groundedness has a rather intuitive motivation: "*it corresponds to discarding the reasoning based on epistemic assumptions, which would enable, for example, to conclude that something is true in the world just by assuming to know it*" (Donini et al, 1997). Domini et al. (Donini et al, 1997) renewed the interest in non-monotonic S5 (and other normal modal logics) by studying their grounded versions. They showed, in particular, that grounded non-monotonic S5 does not collapse with S5. The authors in (Osorio and Navarro, 2003) observed that an expressive fragment of a grounded version of S5 can be captured by a 4-valued logic that later was called MFOUR (the commented fragment consists of sentences with modalities applied only to literals). Gelfond proposes the interpretation of *not a* as $\neg\Box a$. In fact, (Baral, 2003) explains that the definition of stable models by Gelfond and Lifschitz (Gelfond and Lifschitz, 1988) was inspired by this transformation. This line of research was continued and developed in further detail in (Osorio et al, 2005). It has been observed that logics for which their negation operator is constructed by some modal logics (as S5 or MFOUR) using Gelfond's idea correspond to paraconsistent logics (Béziau, 2005). This means that by considering standard monotonic paraconsistent logics one can construct non-monotonic logics, see (Osorio et al, 2006; Osorio et al., 2008).

Recently, in (Osorio et al., 2008) an approach for KR was proposed. It can be used with any paraconsistent logic stronger or equal to C_ω, the weakest paraconsistent logic introduced by da Costa himself. In particular, the authors have made a deep study of the paraconsistent logic G'_3 which is a 3-valued and very expressive logic. G'_3 can define the same class of functions as Lukasiewicz 3-valued logic. In G'_3 we can express very directly the strongest intermediate logic (also known as the Gödel's 3-valued logic G_3). With G'_3 (which is monotonic) one can construct non-monotonic semantics such as the G'_3-stable semantics (Osorio et al, 2006; Osorio et al., 2008). This approach is not restricted to a particular fragment of propositional logics, it is defined for any type of propositional theory. It is important to point out that with the G'_3 semantics, one can express the well known stable semantics (Osorio et al., 2008) (see (Gelfond and Lifschitz, 1988; Gelfond and Lifschitz, 1990) to learn about the stable semantics). G'_3 can also be used to model the well known preferred semantics of argumentation (Nieves et al, 2008). A preliminary implementation of the G'_3 semantics has already been developed (López, 2006). One problem with G'_3-stable semantics is that it can not deal with inconsistent information. This is because G'_3-stable semantics does not take advantage of the paraconsistent nature of G'_3.

This paper can be regarded in two parts.

In the first part, we shall present an overview on how to perform non-monotonic reasoning based on paraconsistent logics. In particular, we concentrate our attention on the G'_3-stable semantics, which is based on paraconsistent logics. We will see that the G'_3-stable semantics is a prominent non-monotonic semantic able to capture the stable model semantics, one of the most successful logic programming semantics of the last years. One interesting result *w.r.t.* the G'_3-stable semantics is that it is able to capture the *preferred semantics* which is one of the most accepted argumentation semantics in argumentation theory. It is worth to comment that argumentation theory is another prominent approach for performing NMR.

In the second part of this paper, we explore two approaches about how to construct non-monotonic semantics based on G'_3 that can deal with inconsistent information. The first approach is based on borrowing ideas from possibilistic logic (Dubois et al., 1994). In particular, we will describe how one can discriminate possibilistic G'_3-stable models by considering their possibilistic degrees. It is worth to comment that, since possibilistic logic is a

suitable approach for capturing uncertain information, we will see that, by considering a possibilistic version of the G'$_3$-stable semantics, one can perform NMR under uncertain information. The second approach stays with G'$_3$ and works with extensions, but takes care of not becoming more inconsistent (so to speak) than the original theory.
The structure of our paper is as follows. In section 2, we summarize some basic concepts and definitions used to understand this paper; we present an overview on how to perform non-monotonic reasoning based on paraconsistent logics; we also present the characterization of a semantics for disjunctive programs, called the p-stable semantics, in terms of the paraconsistent logic G'$_3$; we review a fixed point characterization of *p-stable semantics* for disjunctive programs using classical logic; and finally we review how G'$_3$-stable semantics is useful to model different approaches. In section 3, we review some results around the p-stable semantics in order to represent uncertain, inconsistent and incomplete information and perform reasoning with it. In section 4, we explore two approaches about how to construct non-monotonic semantics based on G'$_3$ that can deal with inconsistent information. Finally in section 5, we present the conclusions.

We assume that the reader has some familiarity with basic logic such as chapter one in (Mendelson, 1987).

2 G'$_3$-stable semantics

This section starts summarizing some basic concepts and definitions used to understand this paper; we review two paraconsistent logics; we also present two characterizations of the G'$_3$-stable semantics for disjunctive programs, the former is a fixed point characterization using classical logic and the latter is in terms of the paraconsistent logic G'$_3$. We also include a theorem that establishes an equivalence between the G'$_3$-stable semantics and the p-stable semantics, only for disjunctive programs. So, we shall use any of the two terminologies when dealing with disjunctive programs. Finally, in this section, we review the expressivity of p-stable semantics.

2.1 Background

A *signature* $\mathcal{L}$ is a finite set of elements that we call *atoms*, or *propositional symbols*. The language of a propositional logic has an alphabet consisting of *proposition symbols*: p_0, p_1, …; *connectives*: $\wedge$, $\vee$, $\leftarrow$, $\neg$; and *auxiliary symbols*: (,), where $\wedge$, $\vee$, $\leftarrow$ are 2-place connectives and $\neg$ is a 1-place connective. Formulas are built up as usual in logic. If F is a formula we will refer to its signature $\mathcal{L}_F$ as the set of atoms that occur in F. A *literal* is either an atom a, called *positive literal*; or the negation of an atom $\neg a$, called *negative literal*.

A *clause* is a formula of the form $H \leftarrow B$ (also written as $B \rightarrow H$), where H and B, arbitrary formulas in principle, are known as the *head* and the *body* of the clause respectively. The body of a clause could be empty, in which case the clause is known as a *fact* and can be noted just by $H\leftarrow$. We define a *disjunctive* clause, as a clause of the form $a_1 \vee \ldots \vee a_k \leftarrow b_1 \wedge \ldots \wedge b_n \wedge \neg b_{n+1} \wedge \ldots \wedge \neg b_{n+m}$ where a_i and b_i are atoms, and $k \geq 1$. The size of such a clause is $k+n+m$. In the case $k=1$ such a clause is a *normal* clause. We will abbreviate a disjunctive clause with the expression $A \leftarrow \mathcal{B}^+ \wedge \neg \mathcal{B}^-$, where A is the set of atoms in the head of the clause, $\mathcal{B}^+$ is the set of atoms that appear without negation in the body of the clause, and $\mathcal{B}^-$ is the set of atoms that appear negated in the body of the clause, that is to say: $A = \{a_1,\ldots, a_k\}$, $\mathcal{B}^+ = \{ b_1,\ldots, b_n\}$, $\mathcal{B}^- = \{b_{n+1},\ldots,b_{n+m} \}$. The symbol $\neg$ before a set of atoms, denotes the conjunction of the negations of the atoms belonging to the set. Given a set of atoms M and a signature $\mathcal{L}$, $\neg$ M'= $\{\neg a \mid a \in \mathcal{L} \setminus M\}$.

Finally, a *program* is a finite set of clauses. If all the clauses in a program are of a certain type, we say the program is also of that type. For instance, a set of arbitrary clauses is an *arbitrary program*, a set of disjunctive clauses is a *disjunctive program*. The *size of a disjunctive program* is the sum of the sizes of the clauses in the program.

2.2 Motivation and logical foundations of the G'$_3$-stable semantics

In (Osorio et al., 2008) the authors discuss and illustrate why paraconsistent logics are also interesting for logic programming semantics. They use an example for that purpose and consider the normal program $P_1 : \{a \leftarrow \neg b\}$. The well known stable semantics (Gelfond and Lifschitz, 1988) of P_1 gives $\{a\}$ as the unique intended model of this program. If we use classical logic we obtain $\{b\}$ as a second model, but this is against the spirit of logic programming. However, it would be convenient to have some semantics, which share several properties with the stable semantics, but are closer to classical logic. For example, let us consider the following program P_2: $\{a \leftarrow \neg b,\ a \leftarrow b,\ b \leftarrow a\}$.

P_2 does not have stable models, but the set $\{a,b\}$ could be considered the intended model for P in classical logic. So, we need logics weaker than classical logic in order to be able to model logic programming. A major mathematical question then arises: Are there other well known semantics such that the unique intended model of program P_1 is $\{a\}$ and the unique intended model of program P_2 is $\{a,b\}$? The answer is "yes" when we consider semantics defined by some paraconsistent logics, at least, the two logics that we consider in this paper: the C_ω logic and the G'$_3$ logic.

The C_ω logic (da Costa, 1963) is defined as positive logic[1] plus the following two axioms: $\boldsymbol{C_\omega 1}$**:** $a \vee \neg a$ and $\boldsymbol{C_\omega 2}$**:** $\neg\neg a \rightarrow a$.

Table 1. Truth tables of connectives in G'$_3$

x	$\neg$x
0	2
1	2
2	0

$\rightarrow$	0	1	2
0	2	2	2
1	0	2	2
2	0	1	2

The G'$_3$ logic is defined as a 3-valued logic with truth values in the domain $\mathcal{D} = \{0,1,2\}$ where *2* is the designated value. The evaluation functions of the logic connectives are then defined as follows: $x \wedge y = \min(x, y)$; $x \vee y = \max(x, y)$; and the $\neg$ and $\rightarrow$ connectives are defined according to the truth tables given in Table 1. (Osorio and Carballido, 2008) present an axiomatization of G'$_3$ that consists of all the axioms of C_ω plus the following four axioms:

E1: $(\neg A \rightarrow \neg B) \leftrightarrow (\neg\neg B \rightarrow \neg\neg A)$ where $F \leftrightarrow G := (F \leftarrow G) \wedge (G \leftarrow F)$

E2: $\neg\neg(A \rightarrow B) \leftrightarrow ((A \rightarrow B) \wedge (\neg\neg A \rightarrow \neg\neg B))$

E3: $((B \wedge \neg B) \wedge (\sim\sim A \wedge \neg A)) \rightarrow A$ where $\sim A := (A \rightarrow (\neg A \wedge \neg\neg A))$

E4: $\neg\neg(A \wedge B) \leftrightarrow (\neg\neg A \wedge \neg\neg B)$

(Osorio and Carballido, 2008) also present a soundness and completeness theorem: a formula can be inferred from the axioms and using modus ponens, if and only if, it takes the designated value *2* for any choice of truth values taken by the atoms in the formula. In other words a formula is a theorem if and only if it is a tautology.

We observe that one of the important features of C_ω and G'$_3$ is that the formula $(\neg a \wedge a) \rightarrow b$ is not a theorem. This fact, a formula and its negation do not entail any arbitrary formula, is what makes these logics paraconsistent.

It turns out that the two paraconsistent logics mentioned in this section are related to a semantics which is an alternative answer to the question posed above. We now turn our attention to this semantics: the *p-stable semantics*. The *p-stable semantics* is based on the following definition of *X-stable model* of an arbitrary program given an arbitrary logic *X*. We use the notation $Q \Vdash_X M$ to denote that M is a classical model of Q and Q proves, in logic X, each atom in M.

Definition 1 (Osorio et al, 2006) Given any logic X and an arbitrary program P, a set of atoms $M \subseteq \mathcal{L}_P$ is a X-stable model of P if $P \cup \neg M' \Vdash_X M$. The semantics defined by these models is called the *X-stable semantics*.

[1] A logic whose connectives do not include any negation (Osorio et al., 2008).

Example 1 Let X be the G'$_3$ logic. Let P be the following program $\{b \leftarrow \neg a,\ a \leftarrow \neg b, p \leftarrow \neg a, p \leftarrow \neg p\}$. It is easy to verify that $\{a, p\}$ and $\{b, p\}$ are G'$_3$-stable models of P. Let us observe that the formula $(\neg a \rightarrow a) \rightarrow a$ is a tautology in G'$_3$.

Now, we review a fixed point characterization of the *p-stable semantics* for disjunctive programs using classical logic, since this kind of characterizations is commonly useful for implementations of a semantics. A preliminary implementation of the p-stable semantics based on this characterization is presented in (López, 2006). Following a similar approach to (Gelfond and Lifschitz, 1988) for the stable semantics, the p-stable semantics uses the *RED(P,M)* reduction defined below. It also uses a fixed-point operator in terms of classical logic. From now on, we shall use the symbol $\models_C$ to denote the consequence relation in classical logic.

Definition 2 (Osorio et al, 2006) Let P be a disjunctive program and M be a set of atoms. We define $RED(P,M) := \{A \leftarrow \mathcal{B}^+ \wedge \neg(\mathcal{B}^- \cap M) \mid A \leftarrow \mathcal{B}^+ \wedge \neg \mathcal{B}^- \in P\}$.

Example 2 (Osorio et al, 2006) Take the following disjunctive program P: $\{b \leftarrow \neg a,\ a \leftarrow \neg b,\ p \leftarrow \neg a,\ p \leftarrow \neg p,\ c \vee a \leftarrow p\}$. Given $M = \{a, p\}$, it follows that *RED(P,M)* is the program: $\{ b \leftarrow \neg a,\ a,\ p \leftarrow \neg a,\ p \leftarrow \neg p,\ c \vee a \leftarrow p\}$.

Definition 3 (Osorio et al, 2006) Let P be a disjunctive program, and M be a set of atoms. We say that M is a *p-stable model* of P if M is a classical model of P and $RED(P,M) \models_C M$.

Example 3 (Osorio et al, 2006) }Let us consider the disjunctive program P of Example 2. We know that given $M = \{a, p\}$, it follows that *RED(P,M)* is the program: $\{ b \leftarrow \neg a,\ a,\ p \leftarrow \neg a,\ p \leftarrow \neg p,\ c \vee a \leftarrow p\}$. Hence, since $RED(P,M) \models_C M$, M is a p-stable model of P, and it is the only one.

Now we present the characterization of the p-stable semantics for disjunctive programs in terms of the paraconsistent logic G'$_3$. This characterization is based on the definition of *X-stable model* of an arbitrary program given an arbitrary logic X. In particular, when the logic X corresponds to the logic G'$_3$ in the Definition 1, we obtain G'$_3$-stable models and the G'$_3$-stable semantics. Let us observe that the condition that *$P \cup \neg M$' proves, in logic X, each atom in M* in Definition 1, is strictly stronger than the condition that *every model of $P \cup \neg M$' in the G'$_3$ logic is a model of M*. To see this, let us look at an example. Let P be the program $\{a \leftarrow \neg b,\ b \leftarrow b\}$. Any G'$_3$ valuation of a and b that makes true the rules of $P \cup \{\neg a\}$, gives b the designated value 2; however $\{b\}$ is not a G'$_3$-stable model of P. The characterization of the p-stable semantics for disjunctive programs in terms of the paraconsistent logic G'$_3$ is described in the following theorem.

Theorem 1 (Osorio et al, 2006) Let P be a disjunctive program. Let M be a set of atoms. M is a p-stable model of P iff M is a G'$_3$-stable model of P.

Let us stress the fact that this theorem *establishes an equivalence between the p-stable semantics and the G'$_3$-stable semantics, only for disjunctive programs*. From now on, we shall use any of the two terminologies when dealing with disjunctive programs.

It is worth to mention that the only p-stable model of program P_2, at the beginning of this subsection 2.2, is $\{a,b\}$, which is also the intended model of this program P_2 in classical logic.

We want to mention that in (Pearce, 1999), Pearce characterized the well known stable semantics defined in (Gelfond and Lifschitz, 1988) for disjunctive programs in terms of an arbitrary intermediate logic[2] X and the definition of the *X-stable model*. This characterization is described in the following theorem.

[2] A logic stronger than or equal to Intuitionistic logic and strictly weaker than Classical logic, such as logic G3.

Theorem 2 (Pearce, 1999) Let X be an intermediate logic. Let P be a disjunctive program. Let M be a set of atoms. M is a stable model of P iff M is a X-stable model of P.

We also remark that the authors of (Osorio et al., 2008) present some results that give conditions under which the concepts of stable and p-stable models agree. They present a translation of a disjunctive program D into a normal program N, such that the p-stable model semantics of N corresponds to the stable semantics of D when restricted to the common language of the theories. Besides, they show that if the size of the program D is n then the size of the program N is bounded by An^2 for a constant A. The relevance of this last result is that it shows that the p-stable model semantics for normal programs is powerful enough to express any problem that can be expressed with the stable model semantics for disjunctive programs.

In order to finish this section, we consider the expressivity of the p-stable semantics. We mention three different approaches for knowledge representation based on this semantics: updates, preferences and argumentation.

In case intelligent agents get new knowledge and this knowledge must be added or updated to their knowledge base, it is important to avoid inconsistencies. An update semantics for update sequences of programs based on p-stable semantics is proposed in (Osorio and Zepeda, 2007).

The concept of preferences is considered a vital component of reasoning with real-world knowledge. In (Osorio and Zepeda, 2008), the authors introduce preference rules which allow us to specify preferences as an ordering among the possible solutions of a problem. Their approach allow us to express preferences for arbitrary programs. They also define a semantics for those programs. The formalism used to develop their work is the p-stable semantics.

The main purpose of argumentation theory is to study the fundamental mechanism humans use in argumentation, and to explore ways to implement this mechanism on computers. Recently, in (Carballido et al., 2009) it was shown that given an argumentation framework, its preferred semantics[3] can be characterized by means of a normal program, such that the preferred extensions of the argumentation framework correspond exactly to the G'_3-stable models of the normal program. This kind of result helps to understand the close relationship between two successful approaches of non-monotonic reasoning: argumentation theory and logic programming with negation as failure.

3 G'_3-stable extensions to handle uncertainty

Uncertain, inconsistent and incomplete information is an unavoidable feature of daily decision-making. In order to deal with uncertain, inconsistent and incomplete information intelligently, we need to be able to represent it and to reason about it. Currently, several proposals for dealing with inconsistent programs have been suggested. One of particular interest is proposed in (de Amo et al., 2002). This approach is based on a family of the so called Logics of Formal Inconsistency (LFIs). A major point of about these logics consists in the internalization of the concepts of consistency and inconsistency inside the object language, see (de Amo et al., 2002; de Amo et al. 2000; Carnielli et al. 2007). These logics are used in (de Amo et al., 2002) as a logical framework to model integrated databases. (de Amo et al., 2002) also presents a method that consists basically in constructing a repaired version of the integrated database where inconsistent information could appear. More closer to our work is the research presented in (Sakama, 1995). Here, the stable semantics was generalized to deal with inconsistent programs. However this approach only allows disjunctive programs. In the approach by (Sakama, 1995) it is not clear which proof theory supports their proposal.

In this section, we will outline some results around the p-stable semantics in order to represent uncertain, inconsistent and incomplete information and perform reasoning with it.

Although the G'_3-stable models semantics (see Definition 1) is interesting, it is explosive and can not handle contradictions. The problem is that in Definition 1, we require M to be a classical 2-valued model of the given theory. We suggest a simple solution to this problem. Our idea is to substitute this condition for a weaker form, one

[3] It is worth mentioning that the preferred semantics is one of the most accepted argumentation semantics in argumentation theory (Bench-Capon and Dunne, 2007)

that avoids making the program more inconsistent (so to speak). As we recognize in the conclusions, we need further research to obtain something more valuable.

In the rest of this subsection we present the details of our construction, which is a revised version of the one presented in (Osorio and Carballido, 2008). First, we will review some basic notions that we will need later.

For a given set L of literals and a program P, we say that L is complete *w.r.t* P if every atom that appears in P, either belongs to L or its negation belongs to L. For a given set of negative literals N, we write P_N to denote the program $P \cup N$. For a given program P, we say that it is literal complete if for every atom x in the language of P, either $P \vdash_{G'3} x$ or $P \vdash_{G'3} \neg x$.

Let us consider an example to explain what we have in mind *w.r.t* a nonmonotic semantics that can deal with inconsistent programs. Suppose that we have a program P consisting of 3 formulas: $\neg a \rightarrow b$, $\neg c$, and c. Then P does not have G'$_3$-stable models.

We are interested in the definition of a semantics that can give the following set of literals as the unique output of P, namely $\{\neg a, b, \neg c, c\}$. For a given program P, let *li* be a function that counts basic inconsistencies as follows: $li(P) := |\{x \in \mathcal{L}_P : P \vdash_{G'3} x , P \vdash_{G'3} \neg x.\}|$.

Definition 4 Let P be any program and let N be a set of negative literals. We said that P_N is a *suitable extension of P* if P_N is literal complete and $li(P)=li(P_N)$. Let L be a set of literals that is complete *w.r.t. P* and let $N := \{\neg x : \neg x \in L\}$. We say that L is *a semi-G'$_3$ stable model* of P if P_N is a suitable extension of P and $P_N \vdash_{G'3} \neg L$.

Following this approach, it turns out that $\{\neg a, b, \neg c, c\}$ is the unique semi-G'$_3$ stable model of our example program given before.

The following result is new and important, because it shows that our semantics is indeed a reasonable generalization of the G'$_3$-stable semantics with respect to normal programs. Informally speaking it says that both the G'$_3$-stable semantics and the semi- G'$_3$ stable agree for normal programs.

Theorem 3 Let P be a normal program and M be a set of atoms. Then the following two properties hold:
(a) if M is a G'$_3$-Stable model of P then $\neg(\mathcal{L}_P \setminus M) \cup M$ is a semi-G'$_3$ stable model of P.
(b) if L is a semi-G'$_3$ stable model of P then $At(L)$ is a G'$_3$-Stable model of P, where $At(L)$ denotes the set of atoms in L.

Proof.
First, note that $li(P)=0$.
We first prove (a). Let M be a G'$_3$-Stable model of P. Let N be the set $\neg(\mathcal{L}_P \setminus M)$. Observe that P_N is literal complete and $li(P_N)=0$. Hence, P_N is a suitable extension of P. (*)
Let L be $N \cup M$. Note that L is a set of literals that is complete w.r.t. P. Moreover, $P_N \vdash_{G'3} \neg L$. (**)
By (*) and (**) L is a semi- G'$_3$ stable model of P. So, $\neg(\mathcal{L}_L \setminus M) \cup M$ is a semi- G'$_3$ stable model of P, as desired.

We now prove (b). Suppose that L is a semi- G'$_3$ stable model of P. L is of the form $N \cup M$, where N is the set of negative literals in L and M is the set of positive literals in L. It is immediate that P_N is a suitable extension of P.

To prove that M (also $At(L)$) is a G'$_3$-Stable model of P we need to show that $P_N \vdash_{G'3} M$ and that M is a standard 2-valued model of P. The first condition follows immediately. Hence, we concentrate on proving the second condition. Note that for every atom x, $x \in M$ iff $P_N \vdash_{G'3} x$. In addition, since P_N is literal complete, we also have that for every atom x, $x \notin M$ iff $P_N \vdash_{G'3} \neg x$. Suppose (in order to obtain a proof by contradiction) that M is not a model of P. Then there exists an atom or a rule r in P that is not modeled by M. If r is an atom (say x) then one can easily verify that $P_N \vdash_{G'3} x$ and $P_N \vdash_{G'3} \neg x$. Hence, $li(P_N) > 0$ and so P_N is not a suitable extension of P, but this is a contradiction. Suppose then that r is rule of the form $\alpha \rightarrow x$. Then M models α but M does not model x. By the second condition, we know that $P_N \vdash_{G'3} \neg x$. Now, since M models α, M models every literal l that appears in α and hence $P_N \vdash_{G'3} l$, so $P_N \vdash_{G'3} \alpha$. Since $P_N \vdash_{G'3} r$, by modus ponens: $P_N \vdash_{G'3} x$. Hence, $li(P_N) > 0$ and so P_N is not a

suitable extension of P, but this is a contradiction. Hence, M is a model of P. Thus, $At(L)$ is a G'$_3$-stable model of P, as desired.

Not much has been done with respect to extending a non-monotonic semantics to deal with inconsistent programs. The semi- G'$_3$ stable semantics is one proposal in this line of research. It has two basic but appealing properties that we claim makes it a reasonable proposal. First, as Theorem 3 shows, it properly generalizes the original G'$_3$-stable semantics, namely, for normal programs both semantics agree. Second, as our example illustrates, the semantics has the non-interference property, namely if a program P is made up of two programs P_1, P_2 that have disjoint languages, then the semantics of P with respect to the language of P_1 corresponds exactly with the semantics of P_1.

4 Uncertain reasoning and p-stable semantics

In this selection, we will outline some results around the p-stable semantics in order to represent uncertain, inconsistent and incomplete information and perform reasoning with it.

Possibilistic logic (Dubois, et al. 1994) is a logic of uncertainty tailored for reasoning under incomplete evidence and partially inconsistent knowledge. In this logic all the formulas are attached by degrees of uncertainty; where, these degrees are quantifications of necessity or possibility of the corresponding possibilistic formulas. Based on ideas of possibilistic logic and stable semantics, in (Nicolas et al., 2006; Nieves et al., 2007), the authors defined a possibilistic logic programming approach which is able to deal with reasoning that is at the same time non-monotonic and uncertain. In this approach, a possibilistic logic normal program P is a finite set of possibilistic normal clauses such that a possibilistic normal clause is a standard normal clause attached by a degree of uncertainty α (α belongs to a complete lattice).

In (Nieves et al., 2007), the authors proposed an alternative approach for capturing the semantics of possibilistic normal logic programs. Since this approach is based on the p-stable semantics (the so called possibilistic p-stable semantics), it is less sensitive (in the sense of inconsistency) than the possibilistic logic programming approach based on the stable semantics.

Like the p-stable semantics, the definition of the possibilistic p-stable considers a reduction which is defined as follows:

Definition 5 (Nieves et al., 2007; Nieves, 2008) Let P be a possibilistic normal program and M a set of atoms. We define $PRED(P,M) := \{(\alpha : a \leftarrow \mathcal{B}^+ \wedge \neg(\mathcal{B}^- \cap M)) \mid (\alpha : a \leftarrow \mathcal{B}^+ \wedge \neg\mathcal{B}^-) \in P\}$.

Observe that the reduction PRED is a straightforward generation of the reduction of the p-stable semantics (see Definition 2).

Now by considering the reduction PRED and the proof theory of possibilistic logic, the *possibilistic p-stable semantics* is defined. In order to define it, we use the projection * which removes either the uncertain degrees attached to a possibilistic atom or to a possibilistic clause, and also the operator $\leq$ which defines a partial order between sets of possibilistic atoms[4].

Definition 6 (Nieves et al., 2007; Nieves, 2008) Let P be a possibilistic normal logic program and M be a set of possibilistic atoms such that M^* is a p-stable model of P^*. We say that M is a possibilistic p-stable model of P if and only if $PRED(P,M^*) \vdash_{PL} M$ and there does not exist a set M' such that the following three conditions holds: $M' \neq M$, $PRED(P,M^*) \vdash_{PL} M'$ and $M \leq M'$.

In this definition, $\vdash_{PL}$ denotes the inference in possibilistic logic. Observe that the definition of the possibilistic p-stable semantics is close related to the p-stable semantics. In fact, M is a possibilistic p-stable model of

[4] $A \leq B \Leftrightarrow A^* \subseteq B^*$, and $\forall x, q_1, q_2, (x, q_1) \in A \wedge (x, q_2) \in B$ then $q_1 \leq q_2$

the possibilistic logic program P if and only if M^* is a p-stable model of the logic program P^* (* denotes a projection which removes the possibilistic degrees of any set of possibilistic atoms or a set of possibilistic logic clauses).

Just as the p-stable semantics is related to stable semantics (also called answer set semantics), the possibilistic p-stable semantics is closely related to the possibilistic semantics defined in (Nicolas et al., 2006; Nieves et al., 2007). For instance, the following proposition shows a relationship between the possibilistic answer set semantics (Nieves et al., 2007) and the possibilistic p-stable semantics.

Proposition 1 (Nieves, 2008) Let P be a possibilistic normal program. If M is a possibilistic answer set of P, then the two following conditions hold: **(a)** M^* is a p-stable model of P^*. **(b)** there exists a possibilistic p-stable mode M' of P such that $M \leq M'$ and $M^* = M'^*$.

An interesting property of the possibilistic p-stable semantics is that this semantics supports a kind of monotony *w.r.t.* the inference under possibilistic logic. In order to enunciate this property, we say that P is *equivalent* to P' *under the possibilistic p-stable semantics* if and only if any possibilistic p-stable model of P is also a possibilistic p-stable model of P' and vice versa.

Proposition 2 (Nieves, 2008) Let P be a possibilistic normal program. If $P \vdash_{PL} (x,\alpha)$ then P is equivalent to $P \cup \{(x, \alpha)\}$ under the possibilistic p-stable semantics.

Prioritizing logic clauses, as it is done in possibilistic logic programming, can be also regarded as a preference relation between rules. In fact, by considering the certainty degrees as *preferences*, two criteria for restoring inconsistent possibilistic knowledge bases were defined in (Nieves, 2008). These criteria are based on the notion of maximal consistent subsets of premises. In other words, one tries to recover the maximal consistent subset of possibilistic clauses from an inconsistent possibilistic program to infer consistent information. For instance, let us consider the following possibilistic logic program P: $\{0.3: a \leftarrow \neg b,\ \ 0.5: b \leftarrow \neg c,\ \ 0.6: c \leftarrow \neg a\}$. Observe that P^* has no answer sets, neither p-stable models; hence, P has no possibilistic answer sets neither possibilistic p-stable models. However, an important property of possibilistic logic, that was proved in (Dubois et al., 1994), is that $P \vdash_{PL} (x,\alpha)$ if and only if $P_\alpha \vdash_{PL} (x,\alpha)$, where $P_\alpha = \{c \mid c \in P \text{ and } n(c) \geq \alpha\}$. This property is a key point to restore consistency of an inconsistent possibilistic knowledge base. In fact, in (Dubois et al., 1994), it was introduced the concept of α-cut. Essentially, possibilistic logic, by using α-cut, deletes the set of possibilistic formulae which are lower than the inconsistent degree of the inconsistent knowledge base. In order to illustrate these ideas, let us consider again the program P, specially let us consider the subprogram P_3 : $\{0.5: b \leftarrow \neg c,\ \ 0.6: c \leftarrow \neg a\}$. All we want to point out is that 3 is the *inconsistent degree* of P (see (Nieves, 2008) for a formal definition of this degree). Now observe that P_3 has a possibilistic answer set which is $\{(c, 0.6)\}$. Observe that by removing the possibilistic clause $0.3: a \leftarrow \neg b$, one can recover a consistent subprogram of P. By lack of space, we do not present the formal details of the α-cut *w.r.t.* possibilistic logic program; however, the interested reader can find in (Nieves, 2008) all the technical details.

As we can see, by considering prioritized logic clauses, as it is done in possibilistic logic programming, one can restore the consistency of possibilistic logic programs.

5 Conclusions

In this paper, we have presented an overview of how to perform non-monotonic reasoning based on paraconsistent logics. In particular, we concentrate our attention on a logic programming semantics, called G'$_3$-stable, which is based on paraconsistent logics. We have seen that the scope of the G'$_3$-stable semantics is at least the same as that of the stable models semantics (Gelfond and Lifschitz, 1988). In fact, we have pointed out that this semantics is able to capture the preferred argumentation semantics which is one of the most accepted argumentation semantics in argumentation theory. It is worth to comment that by considering the relationship between argumentation and logic programming semantics, we can explore the non-monotonic reasoning properties of the argumentation semantics. For instance, since the G'$_3$-stable semantics can be constructed by some paraconsistent logic (Osorio et al. 2006;

Osorio et al., 2008), one can study the non-monotonic reasoning properties of the preferred semantics in terms of these logics.

Also we have described an approach for managing uncertain, incomplete and inconsistent information. This approach has features of *possibilistic logic*. These features allow restoring consistency from an inconsistent possibilistic knowledge base.

The issue of non-monotonic reasoning in the context of inconsistent knowledge bases is still open; however, we have presented enough evidence that suggests that paraconsistent logics could support the definition of robust non-monotonic approaches in order to deal with inconsistent knowledge bases. In fact, we have introduced a generalization of the G'3 stable semantics in order to deal with inconsistent programs.

Acknowledgement

We are grateful to anonymous referees for their useful comments.

References

1. **Baral C.,** *Knowledge Representation, Reasoning and Declarative Problem Solving.* Cambridge University Press, Cambridge, 2003
2. **Bench-Capon T. J. M. and Dunne P. E.,** Argumentation in artificial intelligence. *Artificial Intelligence*, 171 (10-15): 619-641 (2007).
3. **Béziau J.,** Paraconsistent logic from a modal viewpoint. *Journal of Applied Logic*, 3:7-14 (2005).
4. **Carballido J. L., Nieves J. C., and Osorio M.,** Inferring Preferred Extensions by Pstable Semantics. Iberoamerican Journal of Artificial Intelligence (Inteligencia Artificial), 13(41): 38-53 (2009).
5. **Carnielli W. A., Coniglio M., and Marcos J.,** Logics of formal inconsistency. *Handbook of Philosophical Logic*, 14:(15-107), Springer, 2007.
6. **da Costa N. C. A.,** On the theory of inconsistent formal systems (in Portuguese). *PhD thesis*, UFPR Curitiva, 1963.
7. **de Amo S., Carnielli W. A., and Marcos J.,** Formal inconsistency and evolutionary databases. *Logic and Logical Philosophy*, 8(1):115-152 (2000).
8. **de Amo S., Carnielli W. A., and Marcos J.,** A logical framework for integrating inconsistent information in multiple databases. Paper presented at *FOIKS*, Vol.2284 of Lecture Notes in Computer Science, Springer, 2002, 67-84.
9. **Donini F. M., Nardi D., and Rosati R.,** Ground nonmonotonic modal logics. *Logic and Computation*, 7(4) (1997).
10. **Dubois D., Lang J., and Prade H.,** Possibilistic logic. In D. Gabbay, C. J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, Vol. 3 of Nonmonotonic Reasoning and Uncertain Reasoning, Oxford University Press, 1994, 439-513.
11. **Gelfond M. and Lifschitz V.,** The Stable Model Semantics for Logic Programming. In Kowalski R. and Bowen K., editors. Paper presented at *5th Conference on Logic Programming*, MIT Press, 1988, 1070-1080.
12. **Gelfond M. and Lifschitz V.,** Logic Programs with Classical Negation. In D. Warren and P. Szeredi, Editors. Paper presented at *7th Int. Conf. on Logic Programming*, Jerusalem, Israel, MIT Press, 1990, 579-597.
13. **Konolige K.,** On the relation between default and autoepistemic logic. In M. L. Ginsberg, editor, *Readings in Nonmonotonic Reasoning*,. Kaufmann, Los Altos, CA, 1987, 195-226.
14. **López A**., Implementing pstable. Paper presented at *LoLaCOM*, CEUR Vol 220, on line: http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-220/, Apizaco, Tlaxcala, 2006.
15. **McDermott D.,** Nonmonotonic logic II: Nonmonotonic modal theories. *ACM Transactions on Computer Systems*, 29: 33-57 (1982).
16. **McDermott D. and Doyle J.,** Non-monotonic logic I. *Artificial Intelligence*, 13:41-72 (1980).
17. **Mendelson E.,** *Introduction to Mathematical Logic*. Wadsworth, Belmont, CA, third edition, 1987.

18. **Minsky M.,** A framework for representing knowledge. In P. Winston, editor, *The Psychology of Computer Vision*, Mcgraw-Hill, New York, 1975.
19. **Nicolas P., Garcia L., Stéphan I., and Lafèvre C.,** Possibilistic Uncertainty Handling for Answer Set Programming. *Annal of Mathematics and Artificial Intelligence*, 47(1-2):139-181 (2006).
20. **Nieves J. C.,** Modeling arguments and uncertain information --- A non-monotonic reasoning approach. *PhD thesis*, Software Department (LSI), Technical University of Catalonia, 2008.
21. **Nieves J. C., Osorio M., and Cortés U.,** Semantics for possibilistic disjunctive programs. In S. Costantini and R. Watson, editors. Paper presented at *Answer Set Programming: Advances in Theory and Implementation (ICLP-07 Workshop)*, 2007, 271-284.
22. **J. C. Nieves, M. Osorio, and U. Cortés.**, Preferred extensions as stable models. *Theory and Practice of Logic Programming (TPLP)*, 8(4):527-543 (2008).
23. **Osorio M., Arrazola J. R., and Carballido J. L.,** Logical Weak Completions of Paraconsistent Logics. *Journal of Logic and Computation*, doi: 10.1093/log-com/exn015 (2008).
24. **Osorio M. and Carballido J.**, Brief study of G'$_3$ logic. *Journal of Applied Non-Classical Logics*, 18(4):475-499 (2008).
25. **Osorio M. and Navarro J. A.**, Modal logic S52 and FOUR (abstract). In *Proceedings of Annual Meeting of the Association for Symbolic Logic*, 2003.
26. **Osorio M., Navarro J. A., Arrazola J., and Borja V.**, Ground nonmonotonic modal logic S5: New results. *Journal of Logic and Computation*, 15(5):787-813 (2005).
27. **Osorio M., Navarro J. A., Arrazola J. R., and Borja V.,** Logics with Common Weak Completions. *Journal of Logic and Computation*, 16(6):867-890 (2006).
28. **Osorio M. and Zepeda C.,** Update sequences based on minimal generalized pstable models. Paper presented *at MICAI 2007: Advances in Artificial Intelligence, 6th Mexican International Conference on Artificial Intelligence*, Vol. 4827 of Lecture Notes in Computer Science, Springer, 2007, 283-293.
29. **Osorio M. and. Zepeda C.,** Pstable theories and preferences. In Electronic Proceedings of the *18th International Conference on Electronics, Communications, and Computers (CONIELECOMP 2008)*, March, 2008.
30. **Pearce D.,** Stable inference as intuitionistic validity. *Journal of Logic Programming*, 38, 79–91 (1999).
31. **Sakama C. and Inoue K.**, Paraconsistent stable semantics for extended disjunctive programs. *Journal of Logic and Computation*, 5:265-285 (1995).

Mauricio Osorio *He is a titular professor in the Departamento de Computación, Electrónica y Mecatrónica at the Universidad de las Americas Puebla. He belongs to the Sistema Nacional de Investigadores in Mexico. He is co-author of more than 100 papers in Workshops/Conferences/Journals. He has a google page at: http://osoriomauri.googlepages.com/*

Claudia Zepeda *She is professor in the Facultad de Computación, at the Benemérita Universidad Autónoma de Puebla. She belongs to the Sistema Nacional de Invesigadores in Mexico. She has a google page at: http://czepedac.googlepages.com/*

Juan Carlos Nieves *He is a researcher in the Knowledge Engineering and Machine Learning Group in the department of Llenguatges i Sistemes Informàtics (LSI) at the Technical University of Catalonia (UPC). Until today, he has published more than 30 research papers in the computer science area. He has participated as reviewer in several international congresses in Artificial Intelligence and Computer Science.*

José Luis Carballido *He obtained his Doctorate degree in 2009 at the Benemérita Universidad Autónoma de Puebla. He has a Master´sdegree in mathematics from MIT, USA. Since 2007 year, he is part time professor at the Facultad de Computación, at the Benemérita Universidad Autónoma de Puebla and also at the Universidad Politecnica de Puebla in the area of informatics.*