Studying Ideal Semantics via Logic Programming Semantics

Juan Carlos Nieves
Universitat Politècnica de Catalunya
Departament de Lleguatges i Sistemes Informàtics
c/Jordi Girona 1-3, E08034, Barcelona, Spain
Email: jcnieves@lsi.upc.edu

Mauricio Osorio
Universidad de las Américas - Puebla
Depto. de Actuaría, Física y Matemáticas
Sta. Catarina Mártir, Cholula, Puebla, 72820 México
Email: osoriomauri@googlemail.com

***Abstract*—In this paper, we show that by using extensions of the Well-Founded Semantics (WFS) which were defined in terms of rewriting systems, one can characterize ideal sets. We also show that these extensions of the well-founded semantics define argumentation semantics with similar behaviour to the ideal argumentation semantics. On the other hand, we introduce a new logic programming semantics which is able to characterize the ideal sets of an argumentation framework.**

***Keywords*-Abstract Argumentation Semantics, Logic Programming Semantics, Non-monotonic Reasoning.**

I. Introduction

Although several approaches have been proposed for argument theory, Dung's approach presented in [8], is a unifying framework which has played an influential role on argumentation research and Artificial Intelligence (AI). Dung's approach is regarded as an abstract model where the main concern is to find the set of arguments which are considered as acceptable. The strategy for inferring the set of acceptable arguments is based on abstract argumentation semantics.

The kernel of Dung's framework is supported by four abstract argumentation semantics: *stable semantics*, *preferred semantics*, *grounded semantics*, and *complete semantics*. Even thought each of these argumentation semantics represents different patterns of selection of arguments, all of them are based on the basic concept of *admissible sets*. Informally speaking, an admissible set represents a coherent and defendable point of view in a conflict between arguments. The stable, preferred and complete semantics represent a *credulous reasoning approach*. This means that for a given problem these semantics identify a set of possible solutions. Unlike the stable, preferred and complete semantics, the grounded semantics represents a *sceptical reasoning approach*. One of the main features of the grounded semantics is that it is polynomial time computable. This fact has encouraged to use the grounded semantics in real argumentation-based systems [4], [12]. However, a common problem of the grounded semantics is that it is easy to identify a set of argumentation frameworks in which the grounded semantics is empty. Due to this weakness of the grounded semantics, Dung *et al.* [9] have introduced an extension of the grounded semantics which is called *ideal semantics*. Due to the sceptical reasoning properties which satisfy the ideal semantics [1], the ideal semantics has been explored from different points of view [1], [9], [10]. Like the stable, preferred, grounded, complete semantics, the ideal semantics is based on the concept of admissible set.

Since Dung's approach was introduced in [8], it was viewed as a special form of logic programming with *negation as failure*. For instance, in [8] it was proved that the grounded semantics can be characterized by the well-founded semantics (WFS) [13]. Given that the ideal semantics is an extension of the grounded semantics, one can expect that the ideal semantics can be studied by extensions of the well-founded semantics.

One can identify at least two approaches for exploring argumentation semantics from the logic programming perspective. On the one hand, one can find some approaches which are mainly concerned in the operational objectives of the argumentation semantics. This means that the main concern of these approaches is to identify an efficient implementation of the given argumentation semantics [11], [16]. These approaches are mainly based on only one logic programming semantics and different codifications of the argumentation semantics. On the other hand, one can find other approaches which characterize different argumentation semantics by considering a *fixed logic programming specification* of an argumentation framework and *different logic programming semantics* [3], [8]. The last approach possibly is the more adequate approach for exploring the common non-monotonic reasoning properties between argumentation semantics and logic programming semantics than the former one. This observation is based on the fact that one can identify the set of acceptable arguments of a given argumentation framework by considering only the models of a particular logic program.

In this paper, we extended the results of [3], [8] in which the authors consider a fixed logic programming specification of an argumentation framework and different logic programming semantics for capturing different selection-patterns of arguments from an argumentation framework. In particular, we explore some extensions of the well-founded semantics which are able to characterize ideal sets. Moreover, we introduce a new logic programming semantics which is able to characterize the ideal sets of an argumentation framework. For constructing this new logic programming semantics, the p-stable semantics is considered [15]. It is worth mentioning that the p-stable semantics is a logic programming semantics able to characterize the preferred argumentation semantics [3].

Since the ideal sets are characterized by logic programming semantics such WFS and WFS$^+$ which are well-behaved semantics in logic programming [6], the results presented in this paper suggest that:

- the ideal semantics satisfies some expected non-monotonic reasoning properties.
- the less restricted scepticism of the ideal semantics is supported by reasoning inferences such as reasoning by cases and logic consequence.

The rest of the paper is divided as follows: in order to have a self contained document all the necessary background is introduced in §II. In §III, a transformation of an argumentation framework into a normal logic program is presented. In §IV, the main results of this paper are presented. Finally in the last section, our conclusions and future work is outlined.

II. Background

In this section, we first define the syntax of a valid logic program, after that some extensions of the Well-Founded Semantics and the p-stable semantics are introduced. In the last part of this section, we present some basic concepts of argumentation theory.

A. Logic programs: Syntax

A signature $\mathcal{L}$ is a finite set of elements that we call atoms. A literal is an atom, a, or the negation of an atom $not\ a$. Given a set of atoms $\{a_1, \dots, a_n\}$, we write $not\ \{a_1, \dots, a_n\}$ to denote the set of literals $\{not\ a_1, \dots, not\ a_n\}$. A normal clause is of the form: $a_0 \leftarrow a_1, \dots, a_j, not\ a_{j+1}, \dots, not\ a_n$, where a_i is an atom, $0 \leq i \leq n$. When $n = 0$ the normal clause is an abbreviation of $a_0 \leftarrow \top$, where $\top$ is the ever true atom. A normal program is a finite set of normal clauses. Sometimes, we denote a clause C by $a \leftarrow \mathcal{B}^+, \neg\mathcal{B}^-$, where $\mathcal{B}^+$ contains all the positive body literals and $\mathcal{B}^-$ contains all the negative body literals. We also use $body(C)$ to denote $\mathcal{B}^+, \neg\mathcal{B}^-$. When $\mathcal{B}^- = \emptyset$, the clause C is called definite clause. A definite program is a finite set of definite clauses. We denote by $\mathcal{L}_P$ the signature of P, i.e. the set of atoms that occurs in P. Given a signature $\mathcal{L}$, we write $Prog_{\mathcal{L}}$ to denote the set of all the programs defined over $\mathcal{L}$.

B. P-stable semantics

Logic consequence in classic logic is denoted by $\vdash$. Given a set of proposition symbols S and a theory (a set of well-formed formulae) Γ, if $\Gamma \vdash S$ if and only if $\forall s \in S\ \Gamma \vdash s$. When we treat a logic program as a theory, each negative literal $not\ a$ is replaced by $\sim a$ such that $\sim$ is regarded as the classical negation in classic logic. Given a normal program P, if $M \subseteq \mathcal{L}_P$, we write $P \Vdash M$ when: $P \vdash M$ and M is a classical 2-valued model of P (*i.e.* atoms in M are set to true, and atoms not in M are set to false; the set of atoms is a classical model of P if the induced interpretation evaluates P to true).

P-stable semantics is defined in terms of a single reduction which is defined as follows:

Definition 1: [15] Let P be a normal program and M be a set of literals. We define $RED(P, M) := \{l \leftarrow \mathcal{B}^+,\ not\ (\mathcal{B}^- \cap M) | l \leftarrow \mathcal{B}^+,\ not\ \mathcal{B}^- \in P\}$.

Let us consider the set of atoms $M_1 := \{a, b\}$ and the following normal program P_1: $a \leftarrow\ not\ b,\ not\ c.$
$a \leftarrow b.\quad b \leftarrow a.$

We can see that $RED(P_1, M_1)$ is: $a \leftarrow\ not\ b.\quad a \leftarrow b.$
$b \leftarrow a.$

By considering the reduction RED, the *p-stable* semantics for normal programs is defined.

Definition 2: [15] Let P be a normal program and M be a set of atoms. We say that M is a p-stable model of P if $RED(P, M) \Vdash M$. $Pstable(P)$ denotes the set of p-stable models of P.

Let us consider again M_1 and P_1 in order to illustrate the definition. We want to verify whether M_1 is a p-stable model of P_1. First, we can see that M_1 is a model of P_1, *i.e.* $\forall\ C \in P_1$, M_1 evaluates C to true. Moreover, we can see that each atom of M_1 can be proved from $RED(P_1, M_1)$ by using classical inference. Then we can conclude that $RED(P_1, M_1) \Vdash M_1$. Hence, M_1 is a *p-stable model* of P_1.

C. The Well-Founded Semantics and extensions

First of all, we present some definitions *w.r.t.* 3-valued logic semantics.

A partial interpretation based on a signature $\mathcal{L}$ is a disjoint pair of sets $\langle I_1, I_2 \rangle$ such that $I_1 \cup I_2 \subseteq \mathcal{L}$. A partial interpretation is total if $I_1 \cup I_2 = \mathcal{L}$. Given two interpretations $I = \langle I_1, I_2 \rangle$, $J = \langle J_1, J_2 \rangle$, we set $I \leq_k J$ *if, by definition,* $I_i \subseteq J_i$, $i = 1, 2$. Clearly $\leq_k$ is a partial order. When we look at interpretations as sets of literals then $\leq_k$ corresponds to $\subseteq$. A general semantics SEM is a function on Prog$_{\mathcal{L}}$ which associates with every program a partial interpretation.

Given a signature $\mathcal{L}$ and two semantics SEM$_1$ and SEM$_2$, we define SEM$_1 \leq_k$ SEM$_2$ if for every program $P \in$ Prog$_{\mathcal{L}}$, SEM$_1(P) \leq_k$ SEM$_2(P)$.

Definition 3 (SEM): For any logic program P, we define HEAD$(P) = \{a |\ a \leftarrow \mathcal{B}^+,\ \neg\mathcal{B}^-\ \in P\}$ — the set of all head-atoms of P. We also define SEM$(P) = \langle P^{true}, P^{false} \rangle$, where $P^{true} := \{p |\ p \leftarrow \top \in P\}$ and $P^{false} := \{p |\ p \in \mathcal{L}_P \backslash \text{HEAD}(P)\}$.

Now, we define some basic transformation rules for normal logic programs which will be considered for characterizing *WFS*.

Definition 4 (Basic Transformation Rules): [7] A transformation rule is a binary relation on $\text{Prog}_{\mathcal{L}}$. The following transformation rules are called *basic*. Let a program $P \in \text{Prog}_{\mathcal{L}}$ be given.

RED$^+$: This transformation can be applied to P, if there is an atom a which does not occur in HEAD(P). **RED**$^+$ transforms P to the program where all occurrences of $not\ a$ are removed.

RED$^-$: This transformation can be applied to P, if there is a rule $a \leftarrow \top \in P$. **RED**$^-$ transforms P to the program where all clauses that contain $not\ a$ in their bodies are deleted.

Succ: Suppose that P includes a fact $a \leftarrow \top$ and a clause $q \leftarrow body$ such that $a \in body$. Then we replace the clause $q \leftarrow body$ by $q \leftarrow body \setminus \{a\}$.

Failure Suppose that P contains a clause $q \leftarrow body$ such that $a \in body$ and $a \notin HEAD(P)$. Then we erase the given clause.

Loop: We say that P_2 results from P_1 by $Loop_A$ if, by definition, there is a set A of atoms such that: 1. for each rule $a \leftarrow body \in P_1$, if $a \in A$, then $body \cap A \neq \emptyset$, 2. $P_2 := \{a \leftarrow body \in P_1 | body \cap A = \emptyset\}$, 3. $P_1 \neq P_2$.

SUB : If P contains two clauses $a \leftarrow body_1$ and $a \leftarrow body_2$, where $body_1 \subseteq body_2$, then $a \leftarrow body_2$ is removed from P

TAUT Suppose P contains a rule C which has the same atom in its head and its positive body. Then we remove this rule.

LC : Suppose $P \vdash a$ for an atom a. Then we add the rule $a \leftarrow \top$ in P.

Let CS_0 be the rewriting system such that contains the transformation rules: RED^+, RED^-, $Succ$, $Failure$, and $Loop$ and $CS_1 = CS_0 \cup \{SUB, TAUT, LC\}$.

We denote the uniquely determined normal form of a program P with respect to the system $\mathcal{CS}$ by $norm_{\mathcal{CS}}(P)$. Every system $\mathcal{CS}$ induces a semantics $\text{SEM}_{\mathcal{CS}}$ as follows: $\text{SEM}_{\mathcal{CS}}(P) := \text{SEM}(norm_{\mathcal{CS}}(P))$. In order to illustrate the basic transformation rules, let us consider the following example.

Example 1: Let P be the following normal program:

$def(b) \leftarrow not\ def(a)$. $\quad def(b) \leftarrow \top$.
$def(c) \leftarrow def(a)$. $\quad def(c) \leftarrow not\ def(b)$.

Now, let us apply CS_0 to P. Since $def(a) \notin HEAD(P)$, then, we can apply **RED**$^+$ to P. Thus we get: $def(b) \leftarrow \top$. $def(c) \leftarrow not\ def(b)$. $\quad def(c) \leftarrow def(a)$.
Notice that we can apply **RED**$^-$ to the new program, thus we get: $def(b) \leftarrow \top$. $\quad def(c) \leftarrow def(a)$.
Finally, we can apply **Failure** to the new program, thus we get: $def(b) \leftarrow \top$. This last program is the *normal form* of P *w.r.t.* CS_0, because none of the transformation rules from CS_0 can be applied.

WFS is one of the most acceptable semantics in logic programming. It was introduced in [13] and was characterized in terms of rewriting systems in [2]. This characterization is defined as follows:

Theorem 1: [2] CS_0 is a confluent rewriting system. It induces a 3-valued semantics that it is the Well-founded Semantics.

There is an extension of WFS which is called WFS^+, this extension of WFS is a logic programming semantics which like WFS is a well-behaved semantics [6]. WFS^+ was introduced in [5] and characterized in terms of rewriting systems in [7].

Theorem 2: [7] The WFS^+ semantics is induced by the confluent rewriting system CS_1.

The consideration of rewriting systems which are confluent and terminating defines a general methodology for characterizing different logic programming semantics. Observe that the differences between WFS and WFS$^+$ are three rewriting rules, *i.e.* SUB, TAUT, LC. Each rewriting rules defines different levels of inference in a logic programming semantics.

Now we present two other rewriting rules which help to define three extensions of WFS with different levels of inference.

Definition 5: [7]

LLC': Let a be an atom that occurs negatively in a program P and also appears in the head of some rule. Let P_1 be the program that results from P by removing $not\ a$ from every clause of P. Let **Succ*** denote the reflexive and transitive closure of the relation **Succ**. Suppose that P_1 relates to P_2 by **Succ*** and $a \in P_2$. In this case, we add $a \leftarrow \top$ to P.

WK: Let P be a program and suppose the following condition holds: $C_1 \in P$, $C_2 \in P$, C_1 is of the form $a \leftarrow l$ and C_2 is of the form $a \leftarrow not\ l$. Then the *WK* transformation replaces the clauses C_1 and C_2 in P by the single clause $a \leftarrow \top$.

By considering LLC' and *WK*, one can define three extensions of WFS as follows:

Lemma 1: [7] Let $CS_2 := CS_0 \cup \{LLC'\}$, $CS_3 := CS_0 \cup \{WK\}$ and $CS_4 := CS_0 \cup \{LLC', WK\}$.

- CS_2 is a confluent rewriting system. It induces a 3-valued semantics that we call $WFS^{LLC'}$.
- CS_3 is a confluent rewriting system. It induces a 3-valued semantics that we call WFS^{WK}.
- CS_4 is a confluent rewriting system. It induces a 3-valued semantics that we call $WFS^{WK+LLC'}$.

D. Argumentation theory

Now, we define some basic concepts of Dung's argumentation approach. The first one is an argumentation framework. An argumentation framework captures the relationships between arguments.

Definition 6: [8] An argumentation framework is a pair $AF := \langle AR, attacks\rangle$, where *AR* is a finite set of arguments, and *attacks* is a binary relation on *AR*, *i.e. attacks* $\subseteq AR \times AR$.

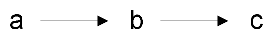

Figure 1. A single argumentation framework

An argumentation framework can be regarded as a directed graph. For instance, if $AF := \langle \{a,b,c\}, \{(a,b),(b,c)\}\rangle$, then AF is represented as in Figure 1. We say that *a attacks b* (or b is attacked by a) if $attacks(a,b)$ holds. Similarly, we say that a set S of arguments attacks b (or b is attacked by S) if b is attacked by an argument in S.

Definition 7: [8]

- A set S of arguments is said to be conflict-free if there are no arguments *a, b* in S such that *a attacks b*.
- An argument $a \in AR$ is acceptable with respect to a set S of arguments if and only if for each argument $b \in AR$: If b attacks a then b is attacked by S.
- A conflict-free set of arguments S is admissible if and only if each argument in S is acceptable *w.r.t.* S.

Definition 8: Let $AF := \langle AR, attacks\rangle$ and $S \subseteq AR$. A set of argument S is:

- preferred if and only if S is a maximal (*w.r.t.* inclusion) admissible set of AF [8].
- ideal if and only if S is admissible and it is contained in every preferred set of AF [9].

III. Mapping from argumentation frameworks to normal programs

In order to see an argumentation framework as a normal program, a mapping from an argumentation framework into a normal logic program is presented. This mapping was introduced in [14]. In this mapping, the predicate $def(x)$, where the intended meaning of $def(x)$ is "x is a defeated argument", is used.

Definition 9: Let $AF := \langle AR, attacks\rangle$ be an argumentation framework and $a \in AR$. We define the transformation function $\Psi(a)$ as follows:

$$\Psi(a) = \bigcup_{b:(b,a)\in attacks}\{def(a) \leftarrow \ not\ def(b)\}\cup \\ \bigcup_{b:(b,a)\in attacks}\{def(a) \leftarrow \bigwedge_{c:(c,b)\in attacks} def(c)\}$$

The direct generalization of the transformation function Ψ to an argumentation framework is defined as follows:.

Definition 10: Let $AF := \langle AR, Attacks\rangle$ be an argumentation framework. We define its associated normal program as follows: $\Psi_{AF} := \bigcup_{a\in AR}\{\Psi_1(a)\}$.

As illustration of this transformation, the normal program P of Example 1 is the associated normal program of the argumentation framework of Figure 1.

Before moving on, we want to observe that by considering Ψ_{AF}, the well-founded semantics and the Pstable semantic, one can characterize the grounded and the preferred semantics respectively [3].

IV. Approaching Ideal Sets

In this section, the main results of this paper are presented. In particular, a set of characterizations of ideal sets in terms of logic programming semantics are presented. Moreover the concept of *ideal model* for normal logic programs is defined. As an application of the ideal models, we show that ideal models are able to characterize the ideal set of an argumentation framework.

We start presenting a set of characterizations of the ideal sets which are based on WFS and its extensions (presented in Section II-C). For this setting, the following notation is introduced. Given an argumentation framework $AF := \langle AR, attacks\rangle$ and $S \subseteq AR$, $f(S) = \{def(a) | a \in S\}$.

Lemma 2: Let $AF := \langle AR, attacks\rangle$ be an argumentation framework and $S, D \subseteq AR$.

1) if $WFS(\Psi_{AF}) = \langle f(D), f(S)\rangle$, then S is an ideal set, denoted by $I^{WFS}(AF)$.
2) if $WFS^{LLC'}(\Psi_{AF}) = \langle f(D), f(S)\rangle$, then S is an ideal set, denoted by $I^{WFS^{LLC'}}(AF)$.
3) if $WFS^{WK}(\Psi_{AF}) = \langle f(D), f(S)\rangle$, then S is an ideal set, denoted by $I^{WFS^{WK}}(AF)$.
4) if $WFS^{WK+LLC'}(\Psi_{AF}) = \langle f(D), f(S)\rangle$, then S is an ideal set, denoted by $I^{WFS^{WK+LLC'}}(AF)$.
5) if $WFS^{+}(\Psi_{AF}) = \langle f(D), f(S)\rangle$, then S is an ideal set, denoted by $I^{WFS^{+}}(AF)$.

Proof: (sketch) (1) follows by Theorem 7 from [3] which proves that WFS and Ψ_{AF} characterize the grounded extensions and the fact that the grounded extension is an ideal set. The reminder points follows by (1) and the following four observations:

1) The 2-valued models of Ψ_{AF} characterize the admissible sets of AF.
2) If $SEM(P_{AF}) = \langle f(D), f(S)\rangle$ such that SEM is either WFS, $WFS^{LLC'}$, WFS^{WK}, $WFS^{WK+LLC'}$ or WFS^{+}, then $\mathcal{L}_{P_{AF}} \setminus f(S)$ is a 2-valued model of Ψ_{AF} and S is an admissible se of AF.
3) The p-stable models of Ψ_{AF} characterizes the preferred extensions of AF.
4) The positive atoms of WFS, $WFS^{LLC'}$, WFS^{WK}, $WFS^{WK+LLC'}$ and WFS^{+} belongs to every p-stable model.

■

Observe that this lemma suggests that the less restricted scepticism of the ideal semantics is supported by reasoning inferences such as reasoning by cases (*i.e.* WK) and local logic consequence (*i.e.* LLC). Let us remember that the transformation rule WS captures the idea of a reasoning by cases and the transformation rule LLC captures the idea of a reasoning based on Local Logic Consequence.

In order to illustrate these characterizations of ideal sets, let us consider the following examples.

Example 2: Let $AF := \langle AR, attacks \rangle$ be an argumentation framework, in which $AR := \{a, b, c\}$ and $attacks := \{(a, a), (a, b), (b, c), (c, b)\}$ (see Figure 2). Hence, Ψ_{AF} is:

$$\begin{array}{ll} def(a) \leftarrow\ not\ def(a). & def(a) \leftarrow def(a). \\ def(b) \leftarrow\ not\ def(a). & def(b) \leftarrow\ not\ def(c). \\ def(b) \leftarrow def(a). & def(b) \leftarrow def(c). \\ def(c) \leftarrow\ not\ def(b). & def(c) \leftarrow def(c), def(a). \end{array}$$

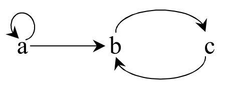

Figure 2. An argumentation framework with two-length cycle and a self-defeated argument.

We can see that the argumentation framework AF has a preferred extension which is $\{c\}$ and the grounded extension is empty. It is obvious that this argumentation framework has two ideal sets: $\{\}$ and $\{c\}$. Now let us see which idea sets are inferred by the argumentation semantics introduced in Lemma 2.

Semantics	3-logic programming model	Ideal Set
$WFS(\Psi_{AF})$	$\langle \{\}, \{\} \rangle$	$\{\}$
$WFS^{LLC'}(\Psi_{AF})$	$\langle \{def(a), def(b)\}, \{def(c)\} \rangle$	$\{c\}$
$WFS^{WK}(\Psi_{AF})$	$\langle \{def(a), def(b)\}, \{def(c)\} \rangle$	$\{c\}$
$WFS^{WK+LLC'}(\Psi_{AF})$	$\langle \{def(a), def(b)\}, \{def(c)\} \rangle$	$\{c\}$
$WFS^{+}(\Psi_{AF})$	$\langle \{def(a), def(b)\}, \{def(c)\} \rangle$	$\{c\}$

Even though in Example 2 all the argumentation semantics introduced in Lemma 2 were able to infer the same ideal set, one can identify different levels of inference with respect to each of those argumentation semantics. Let us consider the following example.

Example 3: Let $AF := \langle AR, attacks \rangle$ be an argumentation framework, in which $AR := \{a, b, c, d\}$ and $attacks := \{(a, b), (b, a), (a, c), (b, c), (c, d), (d, c)\}$ (see Figure 3).

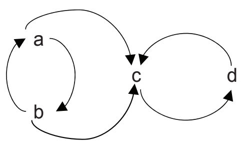

Figure 3. Graphic representation of $AF := \langle \{a, b, c, d\}, \{(a, b), (b, a), (a, c), (b, c), (c, d), (d, c)\} \rangle$.

One can see that the argumentation framework AF has two preferred extensions: $\{a, d\}$ and $\{b, d\}$. On the other hand, we can also see that AF has two ideal sets: $\{\}$ and $\{d\}$. In the following table, we can see the different ideal sets inferred by the well-founded semantics and its extensions.

Semantics	3-logic programming model	Ideal Set
$WFS(\Psi_{AF})$	$\langle \{\}, \{\} \rangle$	$\{\}$
$WFS^{LLC'}(\Psi_{AF})$	$\langle \{,\}, \{\} \rangle$	$\{\}$
$WFS^{WK}(\Psi_{AF})$	$\langle \{def(c), \{\} \rangle$	$\{\}$
$WFS^{WK+LLC'}(\Psi_{AF})$	$\langle \{def(c)\}, \{\} \rangle$	$\{\}$
$WFS^{+}(\Psi_{AF})$	$\langle \{def(c)\}, \{def(d)\} \rangle$	$\{d\}$

As we can see, WFS^{+} is the only logic programming semantics which is able of inferring a non-empty ideal set.

In the following lemma, we define an order between the different ideal sets which were characterized in Lemma 2.

Lemma 3: Let $AF := \langle AR, attacks \rangle$ be an argumentation framework. The following conditions hold:

- $I^{WFS}(AF) \subseteq I^{WFS^{LLC'}}(AF)$, $I^{WFS}(AF) \subseteq I^{WFS^{WK}}(AF)$
- $I^{WFS^{LLC'}}(AF) \subseteq I^{WFS^{WK+LLC'}}(AF)$, $I^{WFS^{WK}}(AF) \subseteq I^{WFS^{WK+LLC'}}(AF)$
- $I^{WFS^{WK+LLC'}}(AF) \subseteq I^{WFS^{+}}(AF)$

Proof: (sketch) The lemma follows by the fact that these order relations are satisfied by WFS, $WFS^{LLC'}$, WFS^{WK}, $WFS^{WK+LLC'}$ and WFS^{+}. ■

Observe that the maximal ideal set that we have characterized in Lemma 2 is gotten by WFS^{+}.

Following the scepticism of the idea semantics from argumentation theory, we introduce the concept of *idea model* of a logic normal program. The construction of ideal models will be based on the p-stable semantics.

Definition 11: Let P be a normal logic program and $M \subseteq \mathcal{L}_P$. M is an ideal model of P if and only if M is a model of P and $\forall M' \in Pstable(P)$, $M' \subseteq M$.

In order to illustrate this definition, let us consider the following example.

Example 4: Let P be the following normal logic program:

$$\begin{array}{ll} def(a) \leftarrow\ not\ def(b). & def(a) \leftarrow def(a). \\ def(b) \leftarrow\ not\ def(a). & def(b) \leftarrow def(b). \end{array}$$

It is easy to see that P has three 2-valued models: $S_1 = \{def(a)\}$, $S_2 = \{def(b)\}$ and $S_3 = \{def(a), def(b)\}$. From these models, one can see that only S_1 and S_2 are p-stable models. Observe that, both S_1 and S_2 are subsets of S_3. This means that S_3 is an ideal model.

An basic property of ideal models *w.r.t.* Ψ_{AF} is that it always has at least an ideal model.

Lemma 4: Let $AF := \langle AR, attacks \rangle$ be an argumentation framework. Ψ_{AF} has always an ideal model which is $\mathcal{L}_{\Psi_{AF}}$

Proof: (sketch) Since every p-stable model M is a subset of $\mathcal{L}_{\Psi_{AF}}$, the lemma follows by the fact that $\mathcal{L}_{\Psi_{AF}}$ always is a model of Ψ_{AF}. ■

An important property of the ideal models is that they are able to characterize ideal sets via the transformation function Ψ.

Theorem 3: Let $AF := \langle AR, attacks \rangle$ be an argumentation framework. S is an ideal model of Ψ_{AF} if and only if $AR \setminus \{a | def(a) \in S\}$ is an ideal set.

Proof: (sketch) The theorem follows by the following observations:

- The 2-valued models of Ψ_{AF} characterize the admissible sets of AF;
- The p-stable models of Ψ_{AF} characterizes the preferred extensions of AF;
- Every ideal model contains every p-stable model of Ψ_{AF}.

■

Let us illustrate this theorem with the following example.

Example 5: Let $AF := \langle AR, attacks \rangle$ be an argumentation framework, where AR := $\{a, b\}$ and attacks := $\{(a, b), (b, a)\}$ (see Figure 4). Ψ_{AF} corresponds to the program P of Example 4. We know that Ψ_{AF} has only one ideal model which is: $\{def(a), def(b)\}$. Hence, AF has only one ideal set which is: $\{\}$.

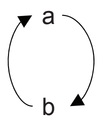

Figure 4. Graphic representation of $AF := \langle \{a, b\}, \{(a, b), (b, a)\} \rangle$.

V. Conclusions and Future Work

We showed that different extensions of the well-founded semantics can characterize different ideal sets (see Lemma 2). From all the ideal sets which were characterized in Lemma 2, I^{WFS^+} is the maximal (*w.r.t.* set inclusion) ideal set which was characterized (see Lemma 3). We conjecture that I^{WFS^+} characterizes the maximal ideal set of an argumentation framework. Observe that if I^{WFS^+} characterizes the maximal ideal set of an argumentation framework, then our approach characterizes the semi-lattice of ideal sets of an argumentation framework. Part of our future work will be to prove these conjectures.

In order to capture the idea of ideal sets in logic programming, we have introduced the concept of ideal model. The construction of ideal models is based on p-stable models. We showed that ideal models define an interesting set of models which characterize the ideal sets of an argumentation framework in a direct way.

Acknowledgment

This work has been partially supported by the CONACyT [CB-2008-01 No.101581].

References

[1] P. Baroni and M. Giacomin. Skepticism relations for comparing argumentation semantics. *Int. J. Approx. Reasoning*, 50(6):854–866, 2009.

[2] S. Brass, U. Zukowski, and B. Freitag. Transformation-based bottom-up computation of the well-founded model. In *NMELP*, pages 171–201, 1996.

[3] J. L. Carballido, J. C. Nieves, and M. Osorio. Inferring Preferred Extensions by Pstable Semantics. *Iberoamerican Journal of Artificial Intelligence (Inteligencia Artificial) ISSN: 1137-3601, (doi: 10.4114/ia.v13i41.1029)*, 13(41):38–53, 2009.

[4] C. I. Chesñevar, A. G. Maguitman, and R. P. Loui. Logical models of argument. *ACM Comput. Surv.*, 32(4):337–383, 2000.

[5] J. Dix. A classification theory of semantics of normal logic programs: I. strong properties. *Fundam. Inform.*, 22(3):227–255, 1995.

[6] J. Dix. A classification theory of semantics of normal logic programs: II. weak properties. *Fundam. Inform.*, 22(3):257–288, 1995.

[7] J. Dix, M. Osorio, and C. Zepeda. A general theory of confluent rewriting systems for logic programming and its applications. *Ann. Pure Appl. Logic*, 108(1-3):153–188, 2001.

[8] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–358, 1995.

[9] P. M. Dung, P. Mancarella, and F. Toni. Computing ideal sceptical argumentation. *Artificial Intelligence*, 171(issues 10-15):642–674, 2007.

[10] P. E. Dunne. The computational complexity of ideal semantics. *Artif. Intell.*, 173(18):1559–1591, 2009.

[11] U. Egly, S. A. Gaggl, and S. Woltran. Aspartix: Implementing argumentation frameworks using answer-set programmin. In M. G. de la Banda and E. Pontelli, editors, *International Conference of Logic Programming (ICLP)*, volume 5366 of *Lecture Notes of Computer Science*, pages 734–738. Springer, 2008.

[12] A. J. García and G. R. Simari. Defeasible logic programming: An argumentative approach. *Theory and Practice of Logic Programming*, 4(1-2):95–138, 2004.

[13] A. V. Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.

[14] J. C. Nieves, M. Osorio, and U. Cortés. Preferred Extensions as Stable Models. *Theory and Practice of Logic Programming*, 8(4):527–543, July 2008.

[15] M. Osorio, J. A. Navarro, J. R. Arrazola, and V. Borja. Logics with Common Weak Completions. *Journal of Logic and Computation*, 16(6):867–890, 2006.

[16] T. Wakaki and K. Nitta. Computing Argumentatoin Semantics in Answer Set Progamming. In *JSAI'2008*, volume 5447 of *Lecture Notes in Computer Science*, pages 254–269, 2009.