Defining Stage Argumentation Semantics in Terms of an Abducible Semantics

Mauricio Osorio[1]

Universidad de las Américas - Puebla
Sta. Catarina Mártir, Cholula, Puebla, México

José Luis Carballido[2] Claudia Zepeda[3]

Benemérita Universidad Autónoma de Puebla
Puebla, Puebla, México

Abstract

We define a new logic programming semantics in terms of abducible atoms. We use it to characterize the stage extensions of an argumentation framework AF by means of an associated normal program P_{AF}. We also define the stage semantics for a special type of normal programs and present a similar characterization.

Keywords: Argumentation semantics, stage argumentation semantics, logic programming semantics.

1 Introduction

Answer Set Programming (ASP) [7] has long being established as a new tool to handle incomplete and inconsistent information. ASP is based on the stable model (answer set) semantics of logic programming [5]. Its scope of applications includes areas like planning, logical agents and artificial intelligence. It is well known that not every logic program has stable models (or answer sets), however the implementation of stable abducible logic programming semantics is intended to fill this gap: by adding facts (abducible atoms) to the original logic program P we can define a new type of models for P. Stable abducible logic programming semantics has been used in different ways, for instance to restore consistency [1]. Following [6,1], we add abducible atoms to a program that does not have answer sets, but whereas for the

---

[1] Email: osoriomauri@gmail.com

[2] Email: jlcarballido7@gmail.com

[3] Email: czepedac@gmail.com

authors of [1] those atoms might not appear in the original language of the program, we choose to select abducible atoms from the 2-valued models of the given program. As some authors have done before [2], we associate a normal logic program P_{AF} to an argumentation framework AF in order to define a semantics of extensions in terms of a logic programming semantics. In our particular case, the programs we use are normal where the clauses have only negative literals in their bodies, and the logic programming semantics we use is based on the stable semantics and the use of abducible atoms, we call it the classical m-ab-m semantics. Our results deal with the stage argumentation semantics [10] and the stage semantics for normal logic programs [8].

The contribution is the following:

- We define a logic programming semantics for a simple kind of normal logic programs called n-programs, in terms of the stable logic programming semantics of augmented programs of the form $P \cup Ab$, where Ab is a family of abducible atoms taken from 2-valued models of P (definition 3.2).

- We characterize the stage logic programming semantics of n-programs (definition 3.14) in terms of the classical stable m-ab-m logic programming semantics just defined (Theorem 3.15). Namely we establish a one to one correspondence between the stage models of P and the classical stable m-ab-m models of P.

- We use this last result to characterize the stage argumentation semantics of an argumentation framework in terms of the stable-abducible logic programming semantics just defined (Corollary 3.17). According to this result the stage argumentation semantics can be defined via a logic programming semantics with abducible atoms.

The rest of the paper is divided as follows: In §2, we present some basic concepts w.r.t. logic programming and argumentation theory. In §3, we characterize the stage argumentation semantics of an argumentation framework based on a stable abducible logic programming semantics. In the last section, we present our conclusions.

2 Background

In this section, we review some theory about logic programming semantics and argumentation semantics. As it has been done before [4,8], we establish the relation between a specific argumentation semantics of an argumentation framework and a semantics defined on the normal logic program associated to the argumentation framework. We start with the basic definitions and facts we need about logic programming semantics and argumentation semantics.

2.1 Logic programming semantics

A signature $\mathcal{L}$ is a finite set of elements that we call atoms. A *literal* is either an atom a, called *positive literal*; or the negation of an atom $\neg a$, called *negative literal*. A *normal* clause is a clause of the form $a \leftarrow b_1 \wedge \ldots \wedge b_n \wedge \neg b_{n+1} \wedge \ldots \wedge \neg b_{n+m}$

where a and each of the b_i are atoms for $1 \leq i \leq n + m$; a is called the *head* and $b_1 \wedge \ldots \wedge b_n \wedge \neg b_{n+1} \wedge \ldots \wedge \neg b_{n+m}$ is called the *body* of the normal clause. If the body of a normal clause is empty, then the clause is known as a *fact* and can be denoted just by: $a \leftarrow$ or $a \leftarrow \top$. We define a *normal logic program P*, as a finite set of normal clauses. By *normal program* we will mean a normal logic program when ambiguity does not arise. We define an *n-program P*, as a normal logic program where every clause is of the form $a \leftarrow \neg b$. We write $\mathcal{L}_P$, to denote the set of atoms that appear in the clauses of P. We want to point out that our negation symbol, $\neg$, corresponds to "*not*" in the standard use of *Logic Programming*.

From now on, we assume that the reader is familiar with the notion of an *interpretation* and *validity* [9]. An interpretation M is called a *2-valued classical model* of P if and only if for each clause $c \in P$, $M(c) = 1$. By *2-valued model* we will mean a 2-valued classical model when ambiguity does not arise.

In this paper, a logic programming semantics S is a mapping defined on the family of all programs which associates to a given program a subset of its 2-valued models. We say that M is a *minimal model* of P if and only if there does not exist a model M' of P such that $M' \subset M$ [9].

2.2   Stable model semantics.

The stable model semantics was defined in terms of the so called *Gelfond-Lifschitz reduction* [5]. Let us recall that a normal positive program always has a unique minimal model.

Definition 2.1 Let P be a normal program. For a set $M \subseteq \mathcal{L}_P$ we define the program P^M obtained from P by deleting each clause that has a literal $\neg l$ in its body with $l \in M$, and then all literals of the form $\neg l$ in the bodies of the remaining clauses. Clearly P^M does not contain the symbol $\neg$. M is a *stable model* of P if and only if M is a minimal model of P^M. We will denote by *stable(P)* the set of all stable models of P.

Example 2.2 For example, let $M = \{b\}$ and $P = \{b \leftarrow \neg a, \ \ c \leftarrow \neg b, \ \ b \leftarrow, \ \ c \leftarrow a\}$. Notice that $P^M = \{b \leftarrow, \ \ c \leftarrow a\}$ has three models: $\{b\}$, $\{b, c\}$ and $\{a, b, c\}$. Since the minimal model among these models is $\{b\}$, we can say that M is a stable model of P.

2.3   Argumentation theory

We review some basic concepts of the *stage argumentation semantics* defined by Verheij [10]. First, we define an argumentation framework.

Definition 2.3 [4] 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., $attacks \subseteq AR \times AR$.

Any argumentation framework can be regarded as a directed graph. For instance, if $AF := \langle \{a, b, c, d\}, \{(a, a), (a, c), (b, c), (c, d)\} \rangle$, then AF is represented as
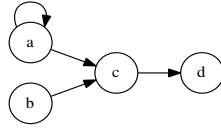
Fig. 1. Graph representation of $AF = \langle\{a, b, c, d\}, \{(a, a), (a, c), (b, c), (c, d)\}\rangle$.

in Figure 1. We say that *a attacks c* (or *c* is attacked by *a*) if $(a, c) \in attacks$ holds. Similarly, we say that a set S of arguments attacks c (or c is attacked by S) if c is attacked by an argument in S.

Intuitively speaking, for a set of arguments to be an extension of an argumentation framework, it is necessary that all of its arguments are consistent in the sense that there are no attacks between them. The next definition states this idea.

Definition 2.4 [4] Let $AF := \langle AR, attacks \rangle$ be an argumentation framework and $S \subseteq AR$. S is said to be *conflict-free* if there are no arguments *a, b* in S such that *a attacks b*.

Now we present the definition of stage argumentation semantics. Given a set S of arguments, $S^+ = \{b \in AR \mid \exists a \in S \text{ such that } (a, b) \in attacks\}$.

Definition 2.5 [10] Let $AF := \langle AR, attacks \rangle$ be an argumentation framework. E is a *stage extension* iff E is a conflict free set and $E \cup E^+$ is maximal with respect to set-inclusion.

Example 2.6 From definition 2.5, it follows that the set $\{b, d\}$ is the only stage extension of the argumentation framework of Figure 1.

2.4 Mapping from argumentation to logic programming

In [3], the authors show that it is possible to characterize semantics in argumentation framework (such as stable extensions or preferred extensions) with semantics in logic programming (such as stable semantics or p-stable semantics) by means of a mapping. The mapping defined in [3] associates a logic program P_{AF} to an argumentation framework AF. In [8], the authors prove that part of the mapping defined in [3] can be used to characterize the stage extensions of an argumentation framework in terms of the stage models of a n-program associated to the framework. In this section, we present that characterization which is defined in terms of the clauses that contain negative literals in their bodies. Hereafter and by abuse of notation, we use P_{AF} to denote this part of the mapping. We use the predicate *d(x)* to represent that "the argument x is defeated", then the clauses such as $d(x) \leftarrow \neg d(y)$ are used to capture the idea that argument x is defeated when anyone of its adversaries y is not defeated.

Definition 2.7 [3] Let $AF = \langle AR, attacks \rangle$ be an argumentation framework, then $P_{AF} = \bigcup_{x \in AR} \{d(x) \leftarrow \neg d(y) \mid (y, x) \in attacks\}$. We say that P_{AF} is the normal program associated to AF.

Example 2.8 Let AF be the argumentation framework of Figure 1. We can see that $P_{AF} = \{\ d(a) \leftarrow \neg d(a),\ d(c) \leftarrow \neg d(a),\ d(c) \leftarrow \neg d(b),\ d(d) \leftarrow \neg d(c)\}$.

Now, we describe the characterization of stage extensions of an argumentation framework AF in terms of the stage models of the normal program P_{AF}. Here, $Facts$ is defined as a function on normal programs which returns the facts which are present in a given normal program.

Definition 2.9 [8] Let $AF = \langle AR, Attacks \rangle$ be an argumentation framework and M be a 2-valued model of P_{AF}. M is a *stage model* of P_{AF} iff $M \setminus Facts((P_{AF})^M)$ is minimal with respect to set inclusion. Observe that $Facts((P_{AF})^M) = (P_{AF})^M$.

We define the following mapping from $2^{\mathcal{L}_{P_{AF}}}$ to 2^{AR} as $E(M) = \{x | d(x) \in (\mathcal{L}_{P_{AF}} \setminus M)\}$.

Theorem 2.10 [8] Let $AF := \langle AR, attacks \rangle$ be an argumentation framework. M is a stage model of P_{AF} iff $E(M)$ is a stage extension of AF.

Example 2.11 We can see that according to example 2.8, the normal program P_{AF} of the argumentation framework AF of Figure 1 has five 2-valued models: $\{d(a), d(b), d(c), d(d)\}$, $\{d(a), d(c), d(d)\}$, $\{d(a), d(b), d(d)\}$, $\{d(a), d(b), d(c)\}$, and $\{d(a), d(c)\}$. Moreover, we can verify that $M = \{d(a), d(c)\}$ is the only *stage model* of P_{AF} since $M \setminus Facts((P_{AF})^M)$ is minimal with respect to set inclusion. Hence, according to Theorem 2.10, $\{b, d\}$ is the only stage extension of AF, which coincides with the result showed in example 2.6.

We remark that definition 2.9 of a stage model can be obtained straightfowardly from the definition of stage model in [8]. In [8], the authors express a stage model in terms of the maximality of a set, for convenience we use a minimality condition on a different set, both definitions are equivalent.

3 Abductive Logic programs and its applications in argumentation

In this section, we use the idea of abducible atom to define a semantics on a very special class of normal programs. The obtained semantics allows us to characterize the stage extensions of an argumentation framework by means of the models of the related normal program defined by this new semantics.

In what follows, the symbol $\subsetneq$ means proper subset. The proofs of the results of this section are presented in Appendix A.

3.1 Abductive logic programs

The classical stable ab-m semantics of a normal program is the particular version of the stable abducible logic programming semantics [6], where the abducibles are taken from 2-valued models of the given program, in opposition to the general case where the atoms are taken from a given arbitrary set.

In this subsection, we define our main semantics for normal programs, called classical m-ab-m semantics, which is used to characterize the stage argumentation semantics of an argumentation framework. Moreover, we describe two properties of the classical m-ab-m semantics which are suitable for that characterization.

In order to define the classical m-ab-m semantics of normal programs, we define a partial order on the set of classical ab-m models and then we find the set of minimal classical ab-m models; these minimal elements define the classical m-ab-m semantics.

We state the following useful fact as an observation.

Observation 1: For any normal program P whose stable semantics is empty, the program $P \cup \mathcal{L}_P$ has $\mathcal{L}_P$ as a stable model. Here, each element of $\mathcal{L}_P$ as a rule in a normal program is interpreted as a fact.

Now, we define the classical stable ab-m semantics of a normal program, where the sets of abducible atoms are taken from 2-valued models of the program.

Definition 3.1 Let M be a 2-valued model of a normal program P and $Ab \subseteq M$. We say that the pair $\langle M, Ab \rangle$ denoted by M_{Ab} is a *classical stable ab-m model* of P if M is a stable model of $P \cup Ab$.

Now, we provide a partial order to the set of classical stable ab-m models and define the classical stable m-ab-m semantics of a normal program as the family of minimal elements in this order. We recall that classical stable m-ab-m semantics plays a central role in this paper, since this semantics will be used to characterize the stage argumentation semantics of an argumentations framework. We also describe two properties of the classical m-ab-m semantics which are suitable for that characterization.

Definition 3.2 Let M_{Ab} be a classical stable ab-m model of a normal program P with $Ab \subseteq M$. We say that the pair M_{Ab} is a *classical stable m-ab-m model* of P if there does not exist another classical stable ab-m model N_{Ab1} such that $Ab1 \subsetneq Ab$.

Example 3.3 Let $P = \{a \leftarrow \neg b, \ b \leftarrow \neg c\}$. We can verify that the only stable model of P is $\{b\}$. Table 1 shows the classical stable ab-m models of P respect to each of the five 2-valued models of P. From Table 1, we can see that the only classical stable m-ab-m model of P is $\langle \{b\}, \{\} \rangle$ and the first entry of this classical stable m-ab-m model of P coincides with the stable model of P. For example, the first row of Table 1 shows the following: Let $M = \{a, c\}$ and let us choose $Ab = \{c\}$ as set of abducibles. Then $(P \cup \{c\})^M = \{a \leftarrow, \ c \leftarrow\}$. Therefore, M is a stable model of $P \cup Ab$. Hence, $\langle \{a, c\}, \{c\} \rangle$ is a classical stable ab-m model of P.

Example 3.4 Let $P = \{a \leftarrow \neg b \ b \leftarrow \neg c, \ c \leftarrow \neg a\}$. We can verify that P does not have stable models. Table 2 shows the classical stable ab-m models of P respect to each of the three 2-valued models of P. From Table 2, we can see that the classical stable m-ab-m semantics of P is: $\{\langle \{a, b\}, \{a\} \rangle, \langle \{b, c\}, \{b\} \rangle, \langle \{a, c\}, \{c\} \rangle\}$.

Now, we state two consequences of observation 1 related to the classical m-ab-m semantics of a normal program.

Table 1
Classical stable ab-m models of P

Model	Abducibles	Classical stable ab-m model
$\{a, c\}$	$\{c\}$	$\langle \{a, c\}, \{c\} \rangle$
$\{b\}$	$\{\}$	$\langle \{b\}, \{\} \rangle$
$\{b, c\}$	$\{b, c\}$	$\langle \{b, c\}, \{b, c\} \rangle$
$\{a, b\}$	$\{a\}$	$\langle \{a, b\}, \{a\} \rangle$
$\{a, b, c\}$	$\{a, b, c\}$	$\langle \{a, b, c\}, \{a, b, c\} \rangle$

Table 2
Classical stable ab-m models of P

Model	Abducibles	classical stable ab-m model
$\{a, c\}$	$\{c\}$	$\langle \{a, c\}, \{c\} \rangle$
$\{b, c\}$	$\{b\}$	$\langle \{b, c\}, \{b\} \rangle$
$\{a, b\}$	$\{a\}$	$\langle \{a, b\}, \{a\} \rangle$

Theorem 3.5 *Let P be a normal program.*

(i) *If the stable semantics of P is not empty, then the classical stable m-ab-m semantics of P coincides with the stable semantics of P.*

(ii) *The classical stable m-ab-m semantics of P is always defined.*

Each item of theorem 3.5 is illustrated by means of examples 3.3 and 3.4 respectively.

Definitions 3.1 and 3.2 colud be extended to more general programs, for example disjunctive programs.

3.2   n-programs

Here, we analyze the classical stable m-ab-m semantics for n-programs. We define a couple of logic programming semantics for n-programs based on abducible atoms: classical stable ab-m semantics and classical normal stable ab-m semantics. Let us remark that the abducibles of these two semantics correspond to subsets of 2-valued models of a given n-program. We also define a partial order for the classical stable ab-m models and the classical normal stable ab-m models in terms of a minimality condition on the sets of abducible atoms, obtaining this way the minimal classical stable ab-m models and the minimal classical normal stable ab-m models (denoted respectively as classical stable m-ab-m models and the classical normal stable m-ab-m models). Finally, we prove that the set of classical stable m-ab-m models and the set of classical normal stable m-ab-m models coincide. In the following subsection, we characterize the stage extensions of an argumentation framework AF based on

classical stable ab-m semantics   classical normal stable ab-m semantics

$\cup$ $\cup$

classical stable m-ab-m semantics $\leftrightarrow$ classical normal stable m-ab-m semantics
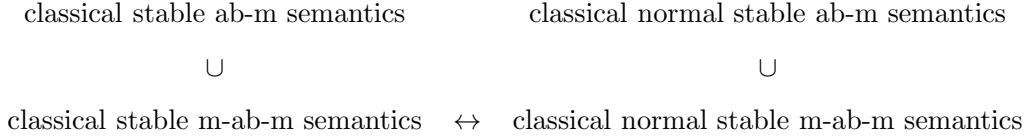
Fig. 2. Relationship among semantics.

the new semantics defined in this subsection when they are applied to the associated n-program P_{AF}. Figure 2 describes the relationship among the semantics defined in this subsection, we use $\cup$ to denote that a semantics is a subset of other semantics, and $\leftrightarrow$ to denote that two semantics coincide.

We recall from section 2 that a n-program is a normal logic program where every clause is of the form $a \leftarrow \neg b$.

Now, we analyze some consequences of definitions 3.1 and 3.2 for the particular case of n-programs.

Lemma 3.6 *Let M be a model of a n-program P and let $Ab \subseteq M$, then M_{Ab} is a classical stable ab-m model of P iff $P^M \cup Ab = M$.*

Note that $(P \cup Ab)^M = P^M \cup Ab$.

Let us remark that in definition 3.2, M and N can be different, so the sets of abducible atoms can be obtained from different sets. A particular case of definition 3.2 is given when the partial order among the classical stable ab-m models of a n-program is defined respect to the set of abducible atoms but these subsets of abducibles are taken from the same set of atoms. This corresponds to the definition of classical normal stable ab-m semantics of a n-program.

Definition 3.7 *Let M_{Ab} be a classical stable ab-m model of a n-program P with $Ab \subseteq M$. We say that the pair $\langle M, Ab \rangle$ is a* classical normal stable ab-m model *of P if there does not exist another classical stable ab-m model M_{Ab1} such that $Ab1 \subsetneq Ab$.*

The next result tell us that, for a fixed model M of P there is always exactly one classical normal stable ab-m model of P.

Lemma 3.8 *Let P be any n-program.*

(i) *For a given M, if M_{Ab1} and M_{Ab2} are two classical stable ab-m models of P, then $M_{(Ab1 \cap Ab2)}$ is a classical stable ab-m model of P.*

(ii) *Classical normal stable ab-m models for P always exist and for a given M a classical normal stable ab-m model M_{Ab} of P is unique.*

(iii) *If M_{Ab} is a classical normal stable ab-m model of P then $P^M \cup Ab = M$ and $P^M \cap Ab = \emptyset$.*

We can define a partial order among the classical normal stable ab-m models of a n-program and also define the concept of minimal classical normal stable ab-m model, called classical normal stable m-ab-m model, based on this partial order.

Table 3
Classical stable ab-m models of P_{AF}

Model	Abducibles	classical stable ab-m model
$\{a, c\}$	$\{a\}$	$\langle \{a, c\}, \{a\} \rangle$
$\{a, b, c\}$	$\{a, b, c\}$	$\langle \{a, b, c\}, \{a, b, c\} \rangle$
$\{a, b, d\}$	$\{a, b\}$	$\langle \{a, b, d\}, \{a, b\} \rangle$
$\{a, c, d\}$	$\{a, d\}$	$\langle \{a, c, d\}, \{a, d\} \rangle$
$\{a, b, c, d\}$	$\{a, b, c, d\}$	$\langle \{a, b, c, d\}, \{a, b, c, d\} \rangle$

Observe that this says that if M_{Ab} is a classical normal stable m-ab-m model of P then P does not have any classical normal stable ab-m model N_{Ab1} with $Ab1 \subseteq Ab$.

Definition 3.9 Let M_{Ab} be a classical normal stable ab-m model of a n-program P with $Ab \subseteq M$. We say that the pair M_{Ab} is a *classical normal stable m-ab-m model* of P if there does not exist another classical normal stable ab-m model of P, N_{Ab1} such that $Ab1 \subsetneq Ab$.

Example 3.10 Let us consider again the program P_{AF} of the argumentation framework AF of Figure 1. Table 3 shows the classical stable ab-m models of P_{AF} respect to each of the five classical 2-valued models of P_{AF}. For convenience, we drop the prefix d of the atoms, for instance, for $d(a)$ we only write a.
From Table 3, we can see the following:

(i) The only classical stable m-ab-m model of P_{AF} is $\langle \{a, c\}, \{a\} \rangle$.

(ii) The classical normal stable ab-m models of P_{AF} are: $\langle \{a, c\}, \{a\} \rangle$, $\langle \{a, b, c\}, \{a, b, c\} \rangle$, $\langle \{a, b, d\}, \{a, b\} \rangle$, $\langle \{a, c, d\}, \{a, d\} \rangle$, and $\langle \{a, b, c, d\}, \{a, b, c, d\} \rangle$.

(iii) The only classical normal stable m-ab-m model of P_{AF} is $\langle \{a, c\}, \{a\} \rangle$.

Finally, we prove that the classical normal stable m-ab-m semantics coincides with the classical stable m-ab-m semantics of a given n-program.

Theorem 3.11 *Let M be a model of a n-program P and $Ab \subseteq M$. M_{Ab} is a classical normal stable m-ab-m model of P iff M_{Ab} is a classical stable m-ab-m model of P.*

In next subsection, we characterize the stage argumentation semantics of an Argumentation Framework AF based on the classical normal stable ab-m logic programming semantics and classical stable m-ab-m logic programming semantics when they are applied to the associated n-program P_{AF}.
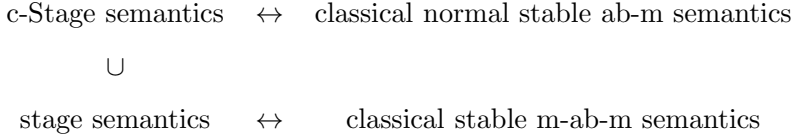
c-Stage semantics $\leftrightarrow$ classical normal stable ab-m semantics

$\cup$

stage semantics $\leftrightarrow$ classical stable m-ab-m semantics

Fig. 3. Relationship among logic programming semantics and stage logic programming semantics.

3.3 Characterization of stage argumentation semantics based on classical stable ab-m semantics

In this section, we define the stage semantics for n-programs. We will see that this semantics corresponds to the stage semantics of an argumentation framework under the mapping presented in definition 5. Next we will show in one of our main results (Theorem 3.15), that for n-programs this semantics can be characterized in terms of the classical stable m-ab-m semantics. As a consequence of this result we prove also that the stage semantics of an argumentation framework AF corresponds to the classical m-ab-m semantics of the associated program P_{AF} (Corollary 3.17). With these results we see that the semantics we have defined in terms of abducible atoms offer an alternative to define the semantics of stage extensions of an argumentation framework as well as the stage semantics for n-programs. The definition of stage logic programming semantics is based on an auxiliar logic programming semantics, called c-Stage. Figure 3 describes the relationship among the logic programming semantics defined in this subsection, we use $\cup$ to denote that a semantics is a subset of other semantics, and $\leftrightarrow$ to denote that two semantics coincide.

Definition 3.12 Let P be a n-program and M be a model of P. We say that the pair $\langle M, (M \setminus P^M) \rangle$ denoted by M_X is a *c-Stage-model* of P. Here $X = M \setminus P^M$.

Theorem 3.13 *Let P be a n-program and M be a model of P. Let X be a set of atoms. Then M_X is classical normal stable ab-m model of P iff M_X is a c-Stage-model of P.*

Definition 3.14 Let P be a n-program and M_X be a c-Stage model of P. We say that the pair M_X is a *stage-model* of P if there does not exist a c-Stage model N_{X1} of P such that $X1 \subsetneq X$.

As we mentioned before, some of the semantics for logic programs as well as for argumentation frameworks can be defined in terms of semantics with abducible atoms. The next result, which is one of our main contributions, shows that for n-programs the stage semantics and the classical stable m-ab-m sematics are equivalent.

Theorem 3.15 *Let P be a n-program, M be a model of P and let $X \subset \mathcal{L}_P$. M_X is a classical stable m-ab-m model of P iff M_X is a stage-model of P.*

Example 3.16 Let us consider again the program P_{AF} of the argumentation framework AF of Figure 1. Table 4 shows the c-Stage models of P_{AF} respect to each of the five classical 2-valued models of P_{AF}.

Table 4
c-Stage models of P_{AF}

Model	c-Stage model
$\{a,c\}$	$\langle\{a,c\},\{a\}\rangle$
$\{a,b,c\}$	$\langle\{a,b,c\},\{a,b,c\}\rangle$
$\{a,b,d\}$	$\langle\{a,b,d\},\{a,b\}\rangle$
$\{a,c,d\}$	$\langle\{a,c,d\},\{a,d\}\rangle$
$\{a,b,c,d\}$	$\langle\{a,b,c,d\},\{a,b,c,d\}\rangle$

From Table 4, we can see the following:

(i) The set of c-Stage models of P_{AF} coincides with the set of classical normal stable ab-m models of P_{AF}, as Theorem 3.13 indicates.

(ii) The only stage-model of P_{AF} is $\langle\{a,c\},\{a\}\rangle$.

(iii) The stage-model $\langle\{a,c\},\{a\}\rangle$ is also the only classical stable m-ab-m model of P_{AF}, as Theorem 3.15 indicates.

We are now ready to present our second result, which is the characterization of the stage extensions of an argumentation framework in terms of the classical stable m-ab-m semantics for n-programs.

Corollary 3.17 *Let $AF := \langle AR, attacks\rangle$ be an argumentation framework and P_{AF} be its associated n-program, then the stage extension semantics of AF is characterized by the classical stable m-ab-m semantics of P_{AF}.*

Example 3.18 Let us take again the argumentation framework AF defined in Figure 1. As we have seen in example 3.16 the only classical stable m-ab-m model of P_{AF} is $\langle\{a,c\},\{a\}\rangle$ which is also the only stage model of P_{AF}. According to Theorem 2.10, the only stage extension of AF is the set $\{b,d\}$ which is the image of the set $\{a,c\}$ under the mapping E (see definition of mapping E in section 2.4). This coincides with the result of example 2.11.

4 Conclusions

We recall the definition of the stage logic programming semantics for a very simple type of normal programs, the n-programs, and we studied the relations between this semantics and the semantics of stage extension for argumentation frameworks. The n-programs we have defined here, besides having a simple structure, have the property that any of them is the associated normal program of an argumentation framework. We characterized both semantics in terms of an abductive logic programming semantics, the classical stable m-ab-m semantics. This suggests the use of the classical stable m-ab-m semantics or possibly another logic programming semantics based on abducible atoms to study extensions of argumentation frameworks

where the associated program is defined by the mapping presented in [3].

Another interesting aspect of the classical stable m-ab-m logic programming semantics and the stage logic programming semantics is that they can be defined for general normal programs, then we can study the relationship between these two semantics in future work.

References

[1] M. Balduccini and M. Gelfond. Logic programs with consistency-restoring rules. In *International Symposium on Logical Formalization of Commonsense Reasoning, AAAI 2003 Spring Symposium Series*, pages 9–18, 2003.

[2] M. Caminada, S. Sá, J. Alcântara, and W. Dvořák. On the equivalence between logic programming semantics and argumentation semantics. *International Journal of Approximate Reasoning*, 58:87–111, 2015.

[3] J. L. Carballido, J. C. Nieves, and M. Osorio. Inferring Preferred Extensions by Pstable Semantics. *Revista Iberomericana de Inteligencia Artificial*, 13(41):38–53, 2009.

[4] 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.

[5] M. Gelfond and V. 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.

[6] A. C. Kakas. Generalized stable models: a semantics for abduction. In *Proc. ECAI'90*, pages 385–391, 1990.

[7] V. Lifschitz. What is answer set programming?. In *AAAI*, volume 8, pages 1594–1597, 2008.

[8] M. Osorio and J. C. Nieves. Range-based argumentation semantics as 2-valued models. *CoRR*, volume abs/1602.08903, 2016.

[9] D. van Dalen. *Logic and structure*. Springer-Verlag, Berlin, 3rd., augmented edition, 1994.

[10] B. Verheij. Two approaches to dialectical argumentation: admissible sets and argumentation stages. *Proc. NAIC*, 96:357–368, 1996.

A   Appendix

Proof. [Theorem 3.5]

(i) If P has stable models, then those models are classical stable ab-m models with the set of abducibles being empty, by definition all these models are classical m-ab-m models.

(ii) We know that the set of atoms in $\mathcal{L}_P$ is a classical model of $P \cup \mathcal{L}_P$. Therefore the classical stable ab-m semantics is not empty and then it follows that the classical stable m-ab-m semantics is not empty.

□

Proof. [Lemma 3.6] Let us assume that M_{Ab} is a classical stable ab-m model of P. Let us split P into two programs $P = \{x_i \leftarrow \neg y_i | y_i \in M\} \cup \{x_j \leftarrow \neg y_j | y_j \notin M\}$.

By assumption M is a minimal model of $(P \cup Ab)^M = \{x_j \leftarrow | y_j \notin M\} \cup \{a_k | a_k \in Ab\}$, which in particular implies that $M = \{x_j\} \cup \{a_k | a_k \in Ab\}$.

Now for the converse, let us assume that $P^M \cup Ab = M$, then $P^M = \{x_j\}$ and we have that $\{x_j\} \cup Ab = M$, hence M is minimal model of $(P \cup Ab)^M$.   □

Proof. [Lemma 3.8]

(i) From Lemma 3.6, we know that $P^M \cup Ab1 = M$ and that $P^M \cup Ab2 = M$, hence it is clear that $P^M \cup (Ab1 \cap Ab2) = M$ and the conclusion follows by the same Lemma 3.6.

(ii) Let M be a model of P, it is clear that M_M is a stable ab-model of P since $P^M \cup M = M$. Then the first result follows from the fact that any partially ordered set has minimal elements. The second part follows from item 1.

(iii) According to Lemma 3.6 if M_{Ab} is a classical normal stable ab-m model of P then $P^M \cup Ab = M$. Let us assume that $m \in P^M \cap Ab$, then $P^M \cup (Ab \backslash \{m\}) = M$ and again by the same Lemma 3.6 $M_{Ab \backslash \{m\}}$ is a classical stable ab-m model of P. This contradicts the minimality of the set Ab.

$\square$

Proof. [Theorem 3.11] The first implication follows from the observation after definition 3.9. Let us assume that M_{Ab} is a classical stable m-ab-m model of P, then there is not $Ab1 \subsetneq Ab$ such that M_{Ab1} is a classical stable ab-m model of P, hence M_{Ab} is a classical normal m-ab model of P, but also by definition, there is not other classical stable m-ab model of P, N_{Ab1} with $Ab1 \subsetneq Ab$. Hence, M_{Ab} is a classical normal stable m-ab-m model of P.

$\square$

Proof. [Theorem 3.13] If M_X is a classical normal stable ab-m model of P then by Lemma 3.8, c) $X = M \backslash P^M$, and then M_X is a c-Stage model of P.

Conversely, if M_X is a c-Stage model of P, then $X = M \backslash P^M$ and then M is a minimal model of $P^M \cup X$, so M_X is a classical normal stable ab-m model of P. $\square$

Proof. [Theorem 3.15] Theorem 3.13 establishes a bijection between the classical normal stable ab-m model of P and the c-Stage models of P. For each of these families the condition that defines the classical stable m-ab-m models and the stage models are equivalent. $\square$

Proof. [Corollary 3.17] Follows from Theorem 2.10 and Theorem 3.15. $\square$