# Expansion and equivalence relations on argumentation frameworks based on logic programs

Juan Carlos Nieves

Department of Computing Science
Umeå University
SE-901 87, Umeå, Sweden
jcnieves@cs.umu.se

**Abstract.** Expansion and equivalence relations have been explored in the settings of abstract argumentation. However, in terms of structured arguments, expansion and equivalence relations have not been explored in the settings of structured arguments based on logic programs. In this paper, we draw connections between resulting argumentation frameworks from logic programs considering expansion and equivalence relations. We show that by considering different methods for constructing arguments and defining attack relations, one can define different expansion and equivalence relations between the resulting argumentation frameworks from logic programs. Moreover, we extended results from abstract argumentation into structured arguments based on logic programs.

## 1 Introduction

Argumentation has been regarded as a non-monotonic reasoning approach since it was suggested as an inference reasoning approach [22]. Dung showed that argumentation inference can be regarded as a logic programming inference with *negation as failure* [10]. Indeed logic programming with negation as failure has been playing an important role in the developments of argumentation. For instance, it has been shown that well-accepted argumentation semantics can be characterized in terms of the inference of logic programming semantics[1]. Moreover, some of the well-performed argumentation solvers are based on logic programming solvers [8]. We can observe that most of these developments have been done in the settings of *abstract argumentation*. This means that these developments consider arguments without an internal structure. Hence the use of these developments in applications which require arguments with an internal structure is not straightforward.

We can argue that depending of the specification language of a knowledge base and the purpose of the arguments, one can define different internal structures of an argument [2, 1, 10, 12, 17]. In the settings of logic programming with negation as failure, one can find different approaches for constructing *structured arguments* [5, 10, 16, 24]. Structured arguments based on logic programs are usually characterized by a tuple of the form $\langle S, C\rangle$ where $S$ is called the *support* of the argument and $c$ is called the *conclusion* of the argument. $S$ is a subset of a logic program, which *derives* the conclusion

[1] A summary of these characterizations can be found in Section 4 of [20]

$c$. The main differences between the different approaches for constructing arguments rely on the conditions which have to satisfy $S$. There are approaches which only ask for syntactic constraints on $S$ [24] and other approaches ask for semantic-based inference conditions on $S$ [16]. Hence, depending on the approach for constructing arguments, one can construct different sets of arguments from the same knowledge base. Moreover, these sets of arguments constructed from a knowledge base will affect both the inferred information from a given knowledge base and the quality of the inferred information[2].

Against this background, we draw connections between resulting argumentation frameworks from logic programs considering expansion and equivalence relations[1, 4, 9, 19]. Given the dynamics of argumentation processes, *e.g.,* dialogues between rational agents [18], equivalence and expansion relations in argumentation have emerged as a relevant research thread in order to compare and relate different argumentation frameworks. In this paper, we focus our attention to a quite common syntactic-based approach for constructing arguments [21, 24] and a semantic-based approach for constructing arguments [16]. We will show that considering the different sets of arguments which can be constructed following these approaches, one can define expansions between argumentation frameworks resulting from a logic program. We will also observe that the way of defining attack relations between arguments has consequences in the structure of the resulting argumentation frameworks. We introduce the property of *sub-argument transitive attack property* which is not fulfilled by the syntactic-based approach for defining attacks between arguments. On the other hand, this property is fulfilled by the semantic-based approach for defining attacks between arguments. In the last part of the paper, we identify a class of logic programs which suggests equivalences in terms of the outputs of the resulting argumentation frameworks.

Let us observe that to the best of our knowledge, the results presented in this paper are the first results which connect structured-based argumentation based on logic programming and expansion relations. It worth mentioning that, in the literature of formal argumentation, expansion and equivalence relations have been explored mainly in the settings of abstract argumentation.

The rest of the paper is split as follows: In §2, a basic background about logic programming and argumentation theory is introduced. In §3, we identify relevant differences of the resulting argumentation frameworks from a logic programming by considering different approaches for constructing arguments. In §4, we show properties of the argumentation frameworks resulting from a logic program *w.r.t.* expansion relations and equivalence relations in terms of outputs. In the last section, we outline our conclusions and future work.

## 2 Background

In this section, a basic background on logic programming and argumentation theory is presented. In terms of logic programming, the class of extended logic programs and the stable model semantics are defined. Regarding argumentation theory, basic definitions

[2] By quality of the inferred information, we mean the satisfaction of conditions such as *consistency* [7].

on argumentation semantics and relations of expansion and equivalence between argumentation frameworks (based on arguments without an internal structure) are presented.

We are assuming that the reader has a basic knowledge on classical logic and logic programming with negation as failure. Indeed, by space limitation, concepts such as interpretation, model, minimal model, stratified logic programs are not defined. For an introduction of these concepts, we encourage the reader to see [3]

### 2.1 Extended logic programs

Let us introduce the language of a propositional logic, which is constituted by propositional symbols: $p_0, p_1, \ldots$; connectives: $\wedge, \leftarrow, \neg,\ not, \top$; and auxiliary symbols: ( , ), in which $\wedge, \leftarrow$ are 2-place connectives, $\neg,\ not$ are 1-place connectives and $\top$ is a 0-place connective. The propositional symbols, the 0-place connective $\top$ and the propositional symbols of the form $\neg p_i\ (i \geq 0)$ stand for the indecomposable propositions, which we call *atoms*, or *atomic propositions*. The atoms of the form $\neg a$ are also called *extended atoms* in the literature. In order to simplify the presentation, we call them atoms as well. The negation symbol $\neg$ is regarded as the so-called *strong negation* in the Answer Set Programming literature [3], and the negation symbol *not* as *negation as failure*. A literal is an atom, *a* (called a positive literal), or the negation of an atom *not a* (called a negative literal). A (propositional) extended normal clause, *C*, is denoted:

$$a \leftarrow b_1 \wedge \cdots \wedge b_j \wedge \mathit{not}\ b_{j+1} \wedge \cdots \wedge \mathit{not}\ b_{j+n} \tag{1}$$

in which $j + n \geq 0$, *a* is an atom, and each $b_i$ $(1 \leq i \leq j + n)$ is an atom. We use the term *rule* as a synonym of *clause* indistinctly. When $j + n = 0$, the clause is an abbreviation of $a \leftarrow \top$ (a *fact*), such that $\top$ is the propositional atom that always evaluates to true. In a slight abuse of notation, we sometimes write the clause (1) as $a \leftarrow \mathcal{B}^+ \wedge \mathit{not}\ \mathcal{B}^-$, in which $\mathcal{B}^+ := \{b_1, \ldots, b_j\}$ and $\mathcal{B}^- := \{b_{j+1}, \ldots, b_{j+n}\}$. An extended logic program *P* is a finite set of extended normal clauses. When $n = 0$, the clause is called an *extended definite clause*. By $\mathcal{L}_P$, we denote the set of atoms which appear in *P*. The handling of strong negation in our logic programs will be done as it is usually done in Answer Set Programming literature [3]. Essentially, each atom of the form $\neg\ a$ is replaced by a new atom symbol $a'$ that does not appear in the language of the program. A program without extended atoms will be called a *normal logic program*. Therefore, we can induce a normal logic program from an extended normal logic program by replacing each extended atom with a new symbol. For instance, let $P$ be the program: $a \leftarrow q;\ \neg q \leftarrow r$, then, by replacing each extended atom with a new atom symbol, we will have: $a \leftarrow q;\ q' \leftarrow r$.

In the literature, different logic programming semantics have been proposed for capturing extended logic programs [14, 13]. In this paper, the stable model semantics is considered in order to build arguments. Stable model semantics is one of the most influential logic programming semantics in the non-monotonic reasoning community and is defined as follows:

**Definition 1.** *[14] Let* P *be a normal logic program. For any set* $S \subseteq \mathcal{L}_P$*, let* $P^S$ *be the definite logic program obtained from* P *by deleting*

**(i)** *each rule that has a formula* $not\ l$ *in its body with* $l \in S$*, and then*
**(ii)** *all formulæ of the form* $not\ l$ *in the bodies of the remaining rules.*

*Hence* S *is a stable model of* P *iff* S *is a minimal model of* $P^S$. $STABLE(P)$ *denotes the set of stable models of* $P$.

### 2.2 Argumentation theory

In this section, we introduce basic concepts on abstract argumentation. To this end, the so called *argumentation frameworks* are introduced. Considering argumentation frameworks, argumentation-based inferences have been defined in terms of *argumentation semantics* [10]. Hence, some well-acceptable argumentation semantics will be defined. In the last part of this section, some definitions about expansion and equivalence relations between argumentation frameworls are defined.

**Argumentation semantics:** We start by defining the basic structure of an argumentation framework (AF).

**Definition 2.** *[10] 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 $\subseteq AR \times AR$*.*

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 $S$. We say that $c$ *defends* $a$ if $b$ attacks $a$ and $c$ attacks $b$.

Let us observe that an AF is a simple structure which captures the conflicts of a given set of arguments. In order to select *coherent points of view* from a set of conflicts of arguments, Dung introduced the so-called *argumentation semantics*. These argumentation semantics are based on the concept of an *admissible set*:

**Definition 3.** *[10]*

- *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.* $S$*.*

Let us introduce some notation in order to define some argumentation semantics. Let $AF := \langle AR, attacks \rangle$ and $S \subseteq AR$. $S^+ = \{b | a \in S \text{ and } (a, b) \in attacks\}$.

From a general point of view, an argumentation semantics $\sigma$ is a function which assigns to an argumentation framework $AF$ a set of sets of arguments denoted by $\mathcal{E}_\sigma(AF)$. Each set of $\mathcal{E}_\sigma(AF)$ is called $\sigma$-extension.

**Definition 4.** *[6, 10, 11] Let* $AF := \langle AR, attacks \rangle$ *be an argumentation framework. An admissible set of argument* $S \subseteq AR$ *is:*

- *a* stable *extension of* $AF$ *(*$S \in \mathcal{E}_{stb}(AF)$*) if* $S$ *attacks each argument which does not belong to* $S$*.*
- *a* preferred *extension of* $AF$ *(*$S \in \mathcal{E}_{pr}(AF)$*) if* $S$ *is a maximal (w.r.t. set inclusion) admissible set of* $AF$ *.*
- *a* complete *extension of* $AF$ *(*$S \in \mathcal{E}_{co}(AF)$*) if each argument, which is acceptable with respect to* $S$*, belongs to* $S$*.*
- *a* grounded *extension of* $AF$ *(*$S \in \mathcal{E}_{gr}(AF)$*) if* $S$ *is a minimal (w.r.t. set inclusion) complete extension.*
- *a* semi-stable *extension of* $AF$ *(*$S \in \mathcal{E}_{ss}(AF)$*) if* $S$ *is a complete extension such that* $S \cup S^{+}$ *is maximal w.r.t. set inclusion.*
- *an* ideal *extension of* $AF$ *(*$S \in \mathcal{E}_{id}(AF)$*) if* $S$ *is contained in every preferred extension of* $AF$*.*

In addition to the argumentation semantics based on admissible sets, there are other approaches for defining argumentation semantics. One of these approaches is the approach based on *conflict-free sets*, *e.g.,* [23]. Considering conflict-free sets, Verheij introduced the so-called *stage semantics*:

**Definition 5.** *Let* $AF := \langle AR, attacks \rangle$ *be an argumentation framework.* $E$ *is a stage extension of* $AF$ *(*$E \in \mathcal{E}_{stg}(AF)$*) if* $E$ *is a conflict free set and* $E \cup E^{+}$ *is maximal w.r.t. set inclusion.*

One can observe that given an argumentation semantics $\sigma$ and an argumentation framework $AF$, $\mathcal{E}_{\sigma}(AF)$ can have more than one $\sigma$-extension. Hence, one can define different status of an given argument *w.r.t.* $\sigma$.

**Definition 6 (Status of arguments).** *[1] Let* $AF := \langle AR, attacks \rangle$ *be an argumentation framework,* $a \in AR$ *and* $\sigma$ *be an argumentation semantics.*

- $a$ *is sceptically accepted w.r.t.* $\sigma$ *iff* $a \in \bigcap_{E \in \mathcal{E}_{\sigma}(AF)} E$*.*
- $a$ *is credulously accepted w.r.t.* $\sigma$ *iff* $a \in \bigcup_{E \in \mathcal{E}_{\sigma}(AF)} E$*.*
- $a$ *is rejected accepted w.r.t.* $\sigma$ *iff* $a \notin \bigcup_{E \in \mathcal{E}_{\sigma}(AF)} E$*.*

**Expansion and corresponding equivalence notions:** The evaluation of equivalence between argumentation frameworks considering different argumentation semantics have been explored by the argumentation community [4, 19]. The following definition introduces some relations of equivalence which have been explored in the settings of abstract argumentation without considering a particular argumentation semantics:

**Definition 7.** *[4] Let* $AF$ *and* $AF'$ *be two argumentation frameworks.* $AF'$ *is an expansion of* $AF = \langle AR, attacks \rangle$ *(denoted by* $AF \preceq_{E} AF'$*) iff* $AF' = \langle AR \cup AR', attacks \cup attacks' \rangle$ *where* $AR \cap AR' = attacks \cap attacks' = \emptyset$*. An expansion is called*

1. *normal (*$AF \preceq_{N} AF'$*) iff* $\forall a, b$ *if* $(a, b) \in attacks'$ *then* $a \in AR'$ *or* $b \in AR'$*.*
2. *strong (*$AF \preceq_{S} AF'$*) iff* $AF \preceq_{N} AF'$ *and* $\forall a, b$ *if* $(a, b) \in attacks'$ *then it does not hold that* $a \in AR$ *and* $b \in AR'$*.*

3. *local* ($AF \preceq_L AF'$) *iff* $AR' = \emptyset$.

Informally speaking, an expansion of an argumentation framework suggests the introduction of new attack relations. These new attack relations can consider new arguments or not. Essentially, a normal expansion introduces new attack relations such that each new attack relation considers new arguments. A strong expansion considers only attacks of new arguments such that the new arguments are not attacked by the original arguments. A local expansion considers new attacks; but, these new attacks are identified considering only the original arguments.

Now let us consider the ideas of equivalence between argumentation frameworks. The following definition introduces different relations of equivalence considering the concepts of expansions and argumentation semantics:

**Definition 8.** *[4] Give an argumentation semantics* $\sigma$*. Two argumentation frameworks* $AF$ *and* $AF'$ *are*

1. *standard equivalence w.r.t.* $\sigma$ ($AF \equiv^{\sigma} AF'$) *iff* $AF$ *and* $AF'$ *possess the same extensions under* $\sigma$*,* i.e. $\mathcal{E}_{\sigma}(AF) = \mathcal{E}_{\sigma}(AF')$.
2. *expansion equivalence w.r.t.* $\sigma$ ($AF \equiv^{\sigma}_{E} AF'$) *iff for each argumentation framework* $AF^*$, $AF \cup AF^* \equiv^{\sigma} AF' \cup AF^*$ *holds,*
3. *normal expansion equivalence w.r.t.* $\sigma$ ($AF \equiv^{\sigma}_{N} AF'$) *iff for each argumentation framework* $AF^*$*, such that* $AF \preceq_N AF \cup AF^*$ *and* $AF' \preceq_N AF' \cup AF^*$, $AF \cup AF^* \equiv^{\sigma} AF' \cup AF^*$ *holds,*
4. *strong expansion equivalence w.r.t.* $\sigma$ ($AF \equiv^{\sigma}_{S} AF'$) *iff for each argumentation framework* $AF^*$*, such that* $AF \preceq_S AF \cup AF^*$ *and* $AF' \preceq_S AF' \cup AF^*$, $AF \cup AF^* \equiv^{\sigma} AF' \cup AF^*$ *holds,*
5. *local expansion equivalence w.r.t.* $\sigma$ ($AF \equiv^{\sigma}_{L} AF'$) *iff for each argumentation framework* $AF^*$*, such that* $AF \preceq_L AF \cup AF^*$ *and* $AF' \preceq_L AF' \cup AF^*$, $AF \cup AF^* \equiv^{\sigma} AF' \cup AF^*$ *holds,*

Unlike expansion relations which are only concern on understanding the new information which added to an argumentation framework, equivalence relations also consider restrictions on how to keep the inferred information from argumentation frameworks considering argumentation semantics.

Let us observe that all the concepts introduced until now are based on abstract arguments. This means that arguments have no a internal structure. In the following section, structured arguments are explored. These structured arguments are constructed from knowledge bases which are expressed in terms of extended normal logic programs.

## 3 Structured arguments

In this section, we explore two approaches for constructing arguments from logic programs. One approach suggests syntactic-based constrains for defining the supports of the suggested arguments [24]. On the other hand, the other approach suggests semantic-based constrains for the supports of the suggested arguments [16]. As we have observed

in Definition 7, attack relations are quite critical for defining expansions of argumentation frameworks. Hence, we introduce an attack relation property which is mainly oriented to structured arguments since this property is based on the idea of sub-arguments. The introduced property is called sub-argument transitive attack property. We will observe that the syntactic-based approach suggested by [24] does not fulfill the so called *sub-argument transitive attack property* (see Proposition 1).

Let us start with the syntactic based approach. The following definition introduces a syntactic-based approach for constructing arguments. As the authors claim in [24], this definition of structured arguments is close related to the suggested definitions by other authors [21].

**Definition 9.** *[24] Let $P$ be a normal logic program. An argument $A$ based on $P$ is a finite tree of rules from $P$ such that:*

1. *each node (of the form $c \leftarrow a_1 \wedge \cdots \wedge a_n \wedge not\ b_1 \wedge \cdots \wedge not\ b_m$ with $n \geq 0$ and $m \geq 0$) has exactly $n$ children, each having a different head $a_i \in \{a_1, \ldots, a_n\}$ $(1 \leq i \leq n)$ and*
2. *no rule occurs more than once in any root-originated branch of the tree.*

*An argument $A$ will be denoted by a tuple of the form $\langle S, c\rangle$ such that $S$ is the set of rules that appear in the tree of $A$ and $c$ is the head of the rule which appears in the root of the tree of $A$. $\mathcal{A}rg_P^1$ denotes the set of all the arguments built from $P$ according to Definition 9.*

Relationships between arguments are defined by the concept of *attack*. Intuitively, an attack between arguments emerges whenever there is a *disagreement* between arguments. Considering the arguments constructed according to Definition 9, the following definition of attack has been defined:

**Definition 10.** *[24] Let $P$ be a normal logic program and $A, B \in \mathcal{A}rg_P^1$ such that $A = \langle S_A, c_A\rangle$ and $B = \langle S_B, c_B\rangle$. We say that $A$ attacks $B$ if $not\ c_A$ appears in $S_B$. $At^1(\mathcal{A}rg_P^1)$ denotes the set of all the attack relationships between the arguments belonging to the set of arguments $\mathcal{A}rg_P^1$.*

Definition 9 follows a syntactic-based approach for constructing the support of arguments. Another option for constructing supports of arguments is to follow a semantic-based approach. In the following definition, the stable model semantic is considered for defining the restrictions of the support of an argument:

**Definition 11.** *[16] Given an extended logic program $P$ and $S \subseteq P$. $Arg_P = \langle S,\ a\rangle$ is an argument under the stable model semantics, if the following conditions hold:*

1. *$S$ is a stratified logic program,*
2. *$a \in M$ such that $M \in STABLE(S)$,*
3. *$S$ is minimal w.r.t. the set inclusion satisfying 2,*
4. *$\nexists\ c \in \mathcal{L}_P$ such that $\{c, \neg c\} \subseteq M$ and $M \in STABLE(S)$.*

By $\mathcal{A}rg_P^2$, we denote the set of all the arguments built from $P$ according to Definition 11. Let us observe that if $\langle S,\ a\rangle \in \mathcal{A}rg_P^2$, then $STABLE(S)$ has exactly one stable model. Moreover, unlike Definition 9 which considers normal logic programs, Definition 11 is considering extended logic programs. It is worth mentioning that $\mathcal{A}rg_P^2$ can be constructed considering Well-Founded-Semantics [13].

From here on, $\mathcal{A}rg_P$ will refer to either $\mathcal{A}rg_P^1$ or $\mathcal{A}rg_P^2$.

One can consider also a semantic-based approach for defining attack relations between arguments.

**Definition 12 (Attack relationship between arguments).** *Let $P$ be an extended logic program. Let $A, B \in \mathcal{A}rg_P$ such that $A = \langle S_A, c_A\rangle$ and $B = \langle S_B, c_B\rangle$. Let $E_A = \bigcap_{M \in STABLE(S_A)} M$ and $E_B = \bigcap_{M \in STABLE(S_B)} M$, we say that $A$ attacks $B$ if one of the following conditions holds:*

1. *$a \in E_A$ and $\neg a \in E_B$.*
2. *$a \in E_A$ and $a \in \mathcal{L}_{S_B} \setminus E_B$.*

*$At^2(\mathcal{A}rg_P)$ denotes the set of all the attack relationships between the arguments belonging to the set of arguments $\mathcal{A}rg_P$.*

Definition 12 identifies attacks between arguments by considering the inferred atoms of each support of the arguments. If there are inconsistencies between the inferred atoms from the supports, attacks between the arguments are defined. The first condition looks for inconsistencies considering strong negation. The second condition looks for inconsistencies considering the semantic interpretation of the atoms.

From here on, $At(\mathcal{A}rg_P)$ will refer to either $At^1(\mathcal{A}rg_P^1)$ or $At^2(\mathcal{A}rg_P)$. As we can observe, Definition 9 and Definition 11 suggest different approaches for constructing arguments. In order to understand the differences between these two approaches for constructing arguments, let us consider the class of normal logic programs which is the class of logic programs in common between the arguments constructed according to Definition 9 and the arguments constructed according to Definition 11.

**Proposition 1.** *Let $P$ be a normal logic program. The following condition holds:*

1. *$\mathcal{A}rg_P^2 \subseteq \mathcal{A}rg_P^1$.*

*Proof.* There are two cases to show:

**a)** If $A \in \mathcal{A}rg_P^2$, then $A \in \mathcal{A}rg_P^1$: If $A \in \mathcal{A}rg_P^2$ and $A = \langle S,\ a\rangle$, then $S$ is a stratified logic program. Since $S$ is minimal (*w.r.t.* set inclusion), then each rule which belongs to $S$ appears only once in $S$. It is direct to see that one can build a three $T$ from $S$ considering Condition 1 of Definition 9. Moreover the root of $T$ is a rule of the form $a \leftarrow \mathcal{B}^+ \wedge not\ \mathcal{B}^-$. Hence, $A \in \mathcal{A}rg_P^1$.

**b)** $\exists A \in \mathcal{A}rg_P^1$ such that $A \notin \mathcal{A}rg_P^2$: Let us suppose that $a \leftarrow\ not\ a \in P$. Then, $\langle \{a \leftarrow\ not\ a\}, a\rangle \in \mathcal{A}rg_P^1$. However, since $STABLE(\{a \leftarrow\ not\ a\}) = \{\}$, $A \notin \mathcal{A}rg_P^2$.

Let us observe that it can be the case that for a given normal logic program $P$, $\mathcal{A}rg_P^2 = \mathcal{A}rg_P^1$ can be true; however, $At^2(\mathcal{A}rg_P^2) \subseteq At^1(\mathcal{A}rg_P^1)$ does not hold for any normal logic program $P$. In order to illustrate this situation, let us consider the following example:

*Example 1.* Let $P$ be a normal logic program with the following set of clauses:

$$n \leftarrow a \qquad p \leftarrow c$$
$$a \leftarrow \ not\ c \qquad c \leftarrow \top$$

We can see that $\mathcal{A}rg_P^1 = \mathcal{A}rg_P^2 = \{Arg_1, Arg_2, Arg_3, Arg_4\}$ where the arguments are defined as follows:

$$Arg_1 = \langle \{n \leftarrow a, a \leftarrow \ not\ c\}, n \rangle$$
$$Arg_2 = \langle \{a \leftarrow \ not\ c\}, a \rangle$$
$$Arg_3 = \langle \{p \leftarrow c, c \leftarrow \top\}, p \rangle$$
$$Arg_4 = \langle \{c \leftarrow \top\}, c \rangle$$

Considering the attack relations suggested by Definition 10, $At^1(\mathcal{A}rg_P^1) = \{(Arg_4, Arg_1), (Arg_4, Arg_2)\}$. On the other hand considering the attack relations suggested by Definition 12, $At^2(\mathcal{A}rg_P^2) = \{(Arg_4, Arg_1), (Arg_4, Arg_2), (Arg_3, Arg_1), (Arg_3, Arg_2)\}$.

In order to understand why $At^2(\mathcal{A}rg_P^2)$ and $At^1(\mathcal{A}rg_P^1)$ are different even that $\mathcal{A}rg_P^2$ and $\mathcal{A}rg_P^1$ can be the same set of arguments, let us introduce the binary relation of *sub-argument*.

**Definition 13 (Sub-argument).** *Let $A = \langle S_A, g_A \rangle$, $B = \langle S_B, g_B \rangle$ be two arguments. $A$ is a sub-argument of $B$ if and only if $S_A \subset S_B$.*

Considering the idea of sub-arguments, the *sub-argument transitive attack property* is defined as follows:

**Definition 14 (Sub-argument transitive attack).** *Let $P$ be a normal logic program and $A, B, C \in At(\mathcal{A}rg_P)$ such that $B$ is a sub-argument of $A$. $At(\mathcal{A}rg_P)$ fulfill sub-argument transitive attack property if the following conditions hold:*

1. *if $B$ attacks $C$, then $A$ attacks $C$.*
2. *if $C$ attacks $B$, then $C$ attacks $A$.*

Considering the arguments introduced by Example 1, we can see that $At^1(\mathcal{A}rg_P^1)$ contains the attack rations in order to satisfy Condition 2 of Definition 14; however, $At^1(\mathcal{A}rg_P^1)$ does not contain the attack relations in order to satisfy Condition 1 of Definition 14. On the other hand, $At^2(\mathcal{A}rg_P^2)$ contains all the attack relations in order to satisfy the property of sub-argument transitive attack. These observations can be expressed in the following proposition:

**Proposition 2.** *Let $P$ be a normal logic program. The following statements hold:*

**a)** *$At^1(\mathcal{A}rg_P^1)$ does not fulfill the property of sub-argument transitive attack.*

**b)** *$At^2(\mathcal{A}rg_P^2)$ fulfills the property of sub-argument transitive attack.*

*Proof.* **a)** The proof is direct by Example 1 which introduces a contra-example.
**b)** Direct by Proposition 5 from [16].

Now that we have defined the concepts of arguments and attacks, let us define the concept of argumentation framework with respect to a logic program as follows:

**Definition 15.** *Let $P$ be a logic program. The resulting argumentation framework w.r.t. $P$ is the tuple: $AF_P = \langle \mathcal{A}rg_P, At(\mathcal{A}rg_P)\rangle$.*

## 4 Expansion relations and equivalence criteria

In this section, we show properties of the argumentation frameworks resulting from a logic program *w.r.t.* expansion and equivalence relations in terms of outputs.

Let us start observing that given a normal logic program $P$, $AF_P^1 = \langle \mathcal{A}rg_P^1, At^1(\mathcal{A}rg_P^1)\rangle$ and $AF_P^2 = \langle \mathcal{A}rg_P^2, At^2(\mathcal{A}rg_P^2)\rangle$, $AF_P^2 \preceq_E AF_P^1$ is false. As we observed in Example 1, $At^2(\mathcal{A}rg_P^2)$ could contain more attack relations than $At^1(\mathcal{A}rg_P^1)$, even though $\mathcal{A}rg_P^2 \subseteq \mathcal{A}rg_P^1$ holds. Hence, given that $\mathcal{A}rg_P^2 \subseteq \mathcal{A}rg_P^1$ holds, an interesting question can be: can we define an expansion for $AF_P^2$ considering $\mathcal{A}rg_P^1$? The answer is yes.

**Proposition 3.** *Let $P$ be a normal logic program, $AF_P^1 = \langle \mathcal{A}rg_P^1, At^1(\mathcal{A}rg_P^1)\rangle$, $AF_P^2 = \langle \mathcal{A}rg_P^2, At^2(\mathcal{A}rg_P^2)\rangle$ and $AF_P^3 = \langle \mathcal{A}rg_P^1, At^1(\mathcal{A}rg_P^1) \cup At^2(\mathcal{A}rg_P^1)\rangle$. The following relations hold:*

**a)** $AF_P^1 \preceq_L AF_P^3$
**b)** $AF_P^2 \preceq_N AF_P^3$

*Proof.* **a)** The proof is direct by the fact that $At^1(\mathcal{A}rg_P^1) \subseteq At^1(\mathcal{A}rg_P^1) \cup At^2(\mathcal{A}rg_P^1)$.
**b)** We start introducing the following notation: $Arg = \mathcal{A}rg_P^1 \setminus \mathcal{A}rg_P^2$ and $At = (At^1(\mathcal{A}rg_P^1) \cup At^2(\mathcal{A}rg_P^1)) \setminus At^2(\mathcal{A}rg_P^2)$.
Not let us introduce the following observations:
**Ob-1:** If $\mathcal{A}rg_P^2 \subseteq \mathcal{A}rg_P^1$, then $At^2(\mathcal{A}rg_P^2) \subseteq At^1(\mathcal{A}rg_P^1) \cup At^2(\mathcal{A}rg_P^1)$.
**Ob-2:** If $\langle S, c\rangle \in Arg$ then either $S$ is not a stratified logic program or $S$ is a stratified logic program but $c \notin \bigcap_{M \in STABLE(S)} M$.
By Proposition 1 and Ob-1, it is direct to see that $AF_P^2 \preceq_E AF_P^3$ is true. By Ob-2, if $A \in Arg$, then the attack relations *w.r.t.* $A$ appear in $At^1(\mathcal{A}rg_P^1)$; hence, if $\exists C \in \mathcal{A}rg_P^1$ such that $C$ attacks $A$ or $A$ attacks $C$, these attack relations belong to $At$. Therefore, $AF_P^2 \preceq_N AF_P^3$.

Let us observe that considering $AF_P^2$ and $AF_P^3$, as they were defined in Proposition 3, it does not hold $AF_P^2 \preceq_S AF_P^3$. In order to illustrate this observation, let us consider the following example:

*Example 2.* Let $P$ be the following logic program:

$a \leftarrow \ not\ a$
$a \leftarrow \ not\ b$
$b \leftarrow \ not\ a$

We can see that $\mathcal{A}rg_P^1 = \{Arg_1, Arg_2, Arg_3\}$ and $\mathcal{A}rg_P^2 = \{Arg_2, Arg_3\}$ such that:

$Arg_1 = \langle\{a \leftarrow \ not\ a\}, a\rangle$
$Arg_2 = \langle\{a \leftarrow \ not\ b\}, a\rangle$
$Arg_3 = \langle\{b \leftarrow \ not\ a\}, b\rangle$

Moreover, $At^1(\mathcal{A}rg_P^1) = \{(Arg1, Arg1), (Arg2, Arg1), (Arg2, Arg3), (Arg3, Arg2)\}$ and $At^2(\mathcal{A}rg_P^2) = \{(Arg2, Arg3), (Arg3, Arg2)\}$. Considering $AF_P^3 = \langle \mathcal{A}rg_P^1, At^1(\mathcal{A}rg_P^1) \cup At^1(\mathcal{A}rg_P^1)\rangle$ as an expansion of $AF_P^2 = \langle \mathcal{A}rg_P^2, At^2(\mathcal{A}rg_P^2)\rangle$. We can see that an argument from $\mathcal{A}rg_P^2$ attacks the new argument argument introduced by $\mathcal{A}rg_P^1$, *i.e.* $Arg_2$ attacks $Arg_1$; hence, $AF_P^3$ cannot be considered as a strong expansion of $\mathcal{A}rg_P^2$. However, $AF_P^3$ is a normal expansion of $AF_P^2$.

Self-loop attacks have been observed as an important condition in the exploration of equivalence [4]. By self-loop attacks, we mean binary relation of the form: an argument $A$ is self-loop attacked if $(A, A) \in attacks$. In [16], it was shown that the resulting argumentation frameworks following a semantics-based approach avoid to contain self-loop attacked arguments. Considering these results, we can show the following relevant theorem:

**Proposition 4.** *Let $P$, $G$ be two extended logic programs and $AF_P = \langle \mathcal{A}rg_P^2, At^2(\mathcal{A}rg_P^2)\rangle$ and $AF_G = \langle \mathcal{A}rg_G^2, At^2(\mathcal{A}rg_G^2)\rangle$. For any $\Phi \in \{E, N, S\}$ and any argumentation semantics $\sigma \in \{stg, stb, ss, pr, id, gr, co\}$:*

$$AF_P = AF_G \textit{ iff } AF_P \equiv_\Phi^\sigma AF_G$$

*Moreover, for $\sigma \in \{stg, ss, pr, id\}$:*

$$AF_P = AF_G \textit{ iff } AF_P \equiv_L^\sigma AF_G$$

*Proof.* Proposition 7 from [16] has shown that given an extended logic program $P$, the resulting argumentation framework $AF_P = \langle \mathcal{A}rg_P^2, At^2(\mathcal{A}rg_P^2)\rangle$ has no arguments which are self-loop attacked. Hence, the proof is direct by Proposition 4.2 from [4].

### 4.1 Equivalence criteria

Amgoud *et al.* [1] have studied equivalence between argumentation systems with structured arguments in terms of *outputs*. In this section, we present results in the study of equivalence regarding outputs. In particular, we identify a class of logic programs which suggests equivalences in terms of outputs. To this end, we extend some concepts introduced by Amgoud *et al.* in order to capture argumentation frameworks constructed from logic programs.

Given a set of arguments $E$, $Base(E) = \bigcup_{\langle S,g\rangle \in E} S$.

**Definition 16 (Outputs).** *Let $AF_P = \langle \mathcal{A}rg_P, At(\mathcal{A}rg_P)\rangle$ be the resulting argumentation argumentation from the extended logic program $P$ and $\sigma$ be an argumentation semantics.*

- $Sc_\sigma(AF_P) = \{A | A \in \mathcal{A}rg_P$ *is sceptical accepted w.r.t.* $\sigma\}$.
- $Cr_\sigma(AF_P) = \{A | A \in \mathcal{A}rg_P$ *is credulously accepted w.r.t.* $\sigma\}$.
- $Output^{sc}_\sigma(AF_P) = \{g_A | A \in \mathcal{A}rg_P$ *such that* $A = \langle S_A, g_A\rangle$ *and* $A$ *is sceptical accepted w.r.t.* $\sigma\}$.
- $Output^{cr}_\sigma(AF_P) = \{g_A | A \in \mathcal{A}rg_P$ *such that* $A = \langle S_A, g_A\rangle$ *and* $A$ *is credulously accepted w.r.t.* $\sigma\}$.
- $Bases_\sigma(AF_P) = \{Base(E) | E \in \mathcal{E}_\sigma(AF_P)\}$.

We introduce our own version of a subset of equivalence criteria introduce by [1].

**Definition 17.** *Let $AF_P = \langle \mathcal{A}rg_P, At(\mathcal{A}rg_P)\rangle$ and $AF_G = \langle \mathcal{A}rg_G, At(\mathcal{A}rg_G)\rangle$ be the resulting argumentation frameworks from the logic programs $P$ and $G$, respectively. Given an argumentation semantics $\sigma$, $AF_P$ and $AF_G$ are equivalent $EQ_i$ ($AF_P \equiv^\sigma_{EQ_i} AF_G$) iff $EQ_i$ holds where $i \in 1, \ldots, 6$ and*

**EQ1** $\mathcal{E}_\sigma(AF_P) = \mathcal{E}_\sigma(AF_G)$,
**EQ2** $Sc_\sigma(AF_P) = Sc_\sigma(AF_G)$,
**EQ3** $Cr_\sigma(AF_P) = Cr_\sigma(AF_G)$,
**EQ4** $Output^{sc}_\sigma(AF_P) = Output^{sc}_\sigma(AF_G)$,
**EQ5** $Output^{cr}_\sigma(AF_P) = Output^{cs}_\sigma(AF_G)$,
**EQ6** $Bases_\sigma(AF_P) = Bases_\sigma(AF_G)$.

Considering the equivalence criteria introduced by Definition 17, syntactic-based arguments and semantics-based arguments, an interesting question is:

> *Is there a class of logic programs which suggests argumentation frameworks which are equivalent in terms of outputs?*

The following proposition identifies a class of logic programs which is an initial answer for the aforementioned question.

**Proposition 5.** *Let $P$ be a stratified normal logic program such that if $a \leftarrow \mathcal{B}^+ \wedge$ not $\mathcal{B}^- \in P$ then $a \notin \mathcal{B}^+$ and $\mathcal{B}^+ \cap \mathcal{B}^- = \emptyset$, $AF^3_P = \langle \mathcal{A}rg^1_P, At^2(\mathcal{A}rg^1_P)\rangle$, $AF^2_P = \langle \mathcal{A}rg^2_P, At^2(\mathcal{A}rg^2_P)\rangle$. The following conditions hold:*

$$AF^1_P \equiv^\sigma_{EQ_i} AF^3_G$$

*where $i \in 1, \ldots, 6$ and $\sigma \in \{stb, pr, co, gr, ss, id, stg\}$.*

*Proof (Sketch).* Let us start observing that if $P$ is a stratified normal logic program such that if $a \leftarrow \mathcal{B}^+ \wedge$ *not* $\mathcal{B}^- \in P$ then $a \notin \mathcal{B}^+$ and $\mathcal{B}^+ \cap \mathcal{B}^- = \emptyset$, then $\mathcal{A}rg^1_P = \mathcal{A}rg^2_P$. Hence, the proof is direct by the fact that $At^2(\mathcal{A}rg^1_P) = At^2(\mathcal{A}rg^2_P)$.

Let us observe that in Proposition 5, both $AF_P^3$ and $AF_P^2$ are considering attack relations which are identified following a semantic-based approach. Moreover, the class of programs which is suggested by Proposition 5 avoids clauses which are tautologies. In this regards, let us observe that the syntactic-based approach for constructing arguments can suggest arguments which their supports contain clauses which are tautologies. On the other hand, the semantic-based approach for constructing arguments does not suggest arguments which their supports contain tautologies.

## 5 Conclusions and future work

Currently there is an intensive research which is mainly oriented to abstract argumentation. However, whenever we consider structured arguments there are different factors which can affect the structure of the resulting argumentation frameworks from a knowledge base.

It is direct to observe that constructing arguments from a logic program considering different (syntactic and semantic) constrains of the supports of these arguments will give place to different argumentation frameworks (Proposition 1 and Proposition 2). Hence to consider different constructions of arguments is not redundant since the resulting argumentation frameworks can infer different information from a given logic program. Moreover, these differences can give place to different strategies for expanding argumentation frameworks (Proposition 3). Let us observe that in a given sequence of *assert moves* in an agent-based dialogue, we are basically expanding argumentation frameworks [18].

It seems that by considering syntactic and semantic restrictions for identifying attack relations, different sets of attack relations can be defined (Proposition 2). We have shown that some properties of structured arguments can help to extend results of abstract argumentation into structured argument as it is the case of self-loop attacks and equivalence relations (Proposition 4). Considering equivalence in terms outputs, we identified a class of logic programs which suggests equivalences in terms of outputs of the resulting argumentation frameworks (Proposition 5).

In the future work, we aim to extend our study considering structured arguments suggested by Dung [10] and the structured arguments suggested by Assumption-Based argumentation ABA [5]. As we have observed with the results of this paper, the way of identifying attack relations affects the final structure of the resulting argumentation frameworks. Moreover, considering different sets of attack relations can define different kind of expansions of the resulting argumentation frameworks from logic programs. Hence, the identification of proper definitions of attack relation is also a goal of our research. It is worth mentioning that in the settings of structured arguments based on classical logic, one can also identify different classes of attacks [15]. However, the definition of these classes of attacks cannot be applied directly in the settings of structured arguments based on logic programming because the inconsistency is defined in other terms *e.g.,* the lack of model.

The identification of classes of logic programs in which different structured argumentation approaches coincide seems to be a relevant issues since these classes of logic

programs can define different algorithms for getting the same outcomes. Hence, this issue is also part of our future work.

## References


1. L. Amgoud, P. Besnard, and S. Vesic. Equivalence in logic-based argumentation. *Journal of Applied Non-Classical Logics*, 24(3):181–208, 2014.
2. L. Amgoud and H. Prade. Using arguments for making and explaining decisions. *Artificial Intelligence*, 173:413–436, 2009.
3. C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, Cambridge, 2003.
4. R. Baumann and S. Woltran. The role of self-attacking arguments in characterizations of equivalence notions. *Journal of Logic and Computation*, 2014.
5. A. Bondarenko, P. M. Dung, R. A. Kowalski, and F. Toni. An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, 93:63–101, 1997.
6. M. Caminada. Semi-Stable semantics. In P. E. Dunne and T. J. Bench-Capon, editors, *Proceedings of COMMA*, volume 144, pages 121–130. IOS Press, 2006.
7. M. Caminada and L. Amgoud. On the evaluation of argumentation formalisms. *Artificial Intelligence*, 171:286–310, 2007.
8. G. Charwat, W. Dvorák, S. A. Gaggl, J. P. Wallner, and S. Woltran. Methods for solving reasoning problems in abstract argumentation - A survey. *Artificial Intelligence*, 220:28–63, 2015.
9. C. I. Chesñevar, G. R. Simari, L. Godo, and T. Alsinet. Expansion operators for modelling agent reasoning in possibilistic defeasible logic programming. In *EUMAS 2005 - Proceedings of the Third European Workshop on Multi-Agent Systems, Brussels, Belgium, December 7-8, 2005*, pages 474–475, 2005.
10. 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.
11. P. M. Dung, P. Mancarella, and F. Toni. Computing ideal sceptical argumentation. *Artificial Intelligence*, 171(10-15):642–674, 2007.
12. M. Elvang-Gøransson, P. Krause, and J. Fox. Acceptability of arguments as 'logical uncertainty'. In *ECSQARU*, pages 85–90, 1993.
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. 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.
15. N. Gorogiannis and A. Hunter. Instantiating abstract argumentation with classical logic arguments: Postulates and properties. *Artif. Intell.*, 175(9-10):1479–1497, 2011.
16. E. Guerrero, J. C. Nieves, and H. Lindgren. Semantic-based Construction of Arguments: an Answer Set Programming Approach. *International Journal of Approximate Reasoning*, 64:54–74, 2015.
17. S. Modgil and H. Prakken. The *ASPIC*$^+$ framework for structured argumentation: a tutorial. *Argument & Computation*, 5(1):31–62, 2014.
18. J. C. Nieves and H. Lindgren. Deliberative argumentation for service provision in smart environments. In *12th European Conference on Multi-Agent Systems (EUMAS 2014)*, LNCS, pages 388–397. Springer, 2015.

19. E. Oikarinen and S. Woltran. Characterizing strong equivalence for argumentation frameworks. *Artif. Intell.*, 175(14-15):1985–2009, 2011.
20. M. Osorio and J. C. Nieves. Range-based argumentation semantics as two-valued models. *Theory and Practice of Logic Programming*, 17(1):75–90, 2017.
21. H. Prakken and G. Sartor. Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics*, 7(1), 1997.
22. H. Prakken and G. A. W. Vreeswijk. Logics for defeasible argumentation. In D. Gabbay and F. Günthner, editors, *Handbook of Philosophical Logic*, volume 4, pages 219–318. Kluwer Academic Publishers, Dordrecht/Boston/London, second edition, 2002.
23. B. Verheij. Two approaches to dialectical argumentation: admissible sets and argumentation stages. In *Proceedings of the Eighth Dutch Conference on Artificial Intelligence (NAIC 1996)*, 1996.
24. Y. Wu, M. Caminada, and D. M. Gabbay. Complete extensions in argumentation coincide with 3-valued stable models in logic programming. *Studia Logica*, 93(2-3):383–403, 2009.