Strawberries and inconsistent ontologies: preferences to the rescue

Master Thesis

Master in *Sciences and Technologies*,
Specialty in *Computer Science*,
Parcours MASTER INFORMATIQUE THÉORIQUE

Auteur
Bruno YUN

Superviseurs
Pierre BISQUERT
Patrice BUCHE
Madalina CROITORU

Lieu de stage
LIRMM - UMR CNRS/UM 5506, Université de Montpellier

version 1 soutenu publiquement le 6 Juin 2016

Abstract

Dans un système d'aide à la décision, certains objectifs sont plus importants que d'autres. Par ailleurs, une information peut provenir d'une source plus sûre qu'une autre. En raison de l'agrégation d'informations de sources diverses dans une base de connaissance, une telle base de connaissances contient souvent des informations qui peuvent être *inconsistantes* entres elles. Pour traiter une base de connaissances ontologique inconsistante, l'approche classique consiste à utiliser des sous-ensembles cohérents maximaux, appelés *réparations*. Nous développerons l'utilité de ces réparations pour l'argumentation dans ce rapport. Nous montrons également comment introduire des *préférences* dans un outil basé sur l'argumentation afin de choisir une solution parmi un ensemble d'alternatives. L'implémentation attendue consiste à définir une interface utilisateur basée sur l'argumentation modélisant les arguments et les attaques entre eux, afin de permettre à l'utilisateur de mieux comprendre pourquoi certaines options peuvent ou ne peuvent pas être acceptées ensemble. La principale nouveauté de ce travail est d'introduire des préférences pour de telles interfaces aussi bien au niveau théorique que pratique. Finalement, dans le cadre du projet INRA Pack4Fresh, la méthode et l'outil seront évalués dans une application d'aide à la décision pour le choix des emballages de fraises afin de réduire le gaspillage (les fraises périmées jetées à la poubelle en raison d'emballages inadaptés) en nous basant sur les connaissances du comportement des consommateurs, des évolutions socio-économiques et des propriétés techniques des emballages à notre disposition.

Abstract

In a decision support system, some goals are more important than others. Moreover, one piece of information can be more certain than another. Due to the fact that information comes from different sources, such a knowledge base can be *inconsistent*. When dealing with an inconsistent ontological knowledge base, the standard approach is to use its maximal consistent subsets, called *repairs*. We will deal with the utility of these repairs for argumentation in this study. We also show how to introduce *preferences* in an argumentation-based tool in order to select a solution among a set of alternatives. The expected implementation consists in defining an argumentation-based user interface by modeling arguments and attacks between them in order to allow the user to better understand why certain options can or cannot be accepted together. The main novelty of this work is to introduce preferences for such interfaces both theoretically and practically. Lastly, in the framework of the INRA project Pack4Fresh, the method and the tool will be analyzed in a decision making software for choosing strawberry packaging in order to reduce waste (people throwing strawberries away because they went off due to unsuited packagings) by linking together consumer behavior insights, socio-economic developments and technical properties of packagings.

Contents

1

Introduction

Argumentation Communication is a pillar of our society, humans have always been concerned with debating and arguing as it constitutes a great part of our daily interactions. Argumentation dialogues are of important effect on our lives as it is implied in debates and decision-making. It is within such argumentation dialogues that opinions of different stakeholders are confronted against each other and arguments are advanced to support them. Dung has concluded in [18] that this process is summarized in a very simple principle: "The one who has the last word laughs best". Abstract argumentation frameworks defined in [18] are systems within which arguments are abstract entities related with an abstract concept of attack relation. One can then extract the several *coherent viewpoints* from an argumentation framework called *extensions* (sets of conflict-free arguments collectively defending each others). For a more detailed formalization of arguments other than the one of abstract argumentation, one can take the road of *structured argumentation* where the construction of arguments is based on a *formal language*. In this approach, arguments have a specific structure and attack are defined with respect to this structure. We chose to consider $Datalog^{\pm}$ as the formal language as it is used a lot in semantics web applications and especially for tractable ontology querying.

Structured Argumentation In this thesis, we will start with two state of the art possibilities to instantiate a logical argumentation framework with $Datalog^{\pm}$. A possibility is to instantiate via $Datalog^{\pm}$ and was proposed in [17, 16]. Another possibility is to use logically instantiated argumentation system such as $ASPIC^{+}$framework. The latter was introduced by *Prakken* in [11, 22, 23, 29, 20] and was proposed in order to formalize the structure of arguments and/or nature of attacks. $ASPIC^{+}$arguments can be composed of other arguments and attacks between them are based on *undercutting, undermining and rebutting*. These three notions represent the different types of conflicts that can occur between two arguments, i.e. conflict on the conclusion (rebutting attack), on the reasoning itself (undercutting attack), or on the hypothesis (undermining attack).

Decision-making Argumentation can also be used to make decisions. In the event that the argumentation system returns more than one extension, it is often difficult to select one out of many alternatives. Many researchers have studied this problem and proposed various ideas. In [4], the authors suggested to vote on extensions but we will not expand on this idea as it is outside of my domain of expertise. Another idea introduced in [19, 16] was to use preferences on

pieces of information that are used to generate the arguments. These preferences can represent either the importance or the confidence of the information and are usually gathered from experts. We chose to focus on preferences as they are widely studied in the field of argumentation and constitute a simple and comprehensive way to explain decisions to users.

Preferences After instantiating an argumentation framework, one can choose to add preferences to refine the output of the framework. Preferences can either occur in the computation of extensions or in the refining of the solutions as described in [5]. In [16], the preferences are viewed as a binary relation $\geq$ on facts (not necessarily total). In such a case, we do not change the computation of extensions and only extract different subset of extensions (locally optimal, Pareto optimal and globally optimal extensions) from the extensions produced. It was shown that this preference-based argumentation system satisfies rationality postulates [17]. Preferences in $ASPIC^{+}$are viewed as a binary ordering on arguments (constructed using an ordering on ordinary premises and defeasible rules) and they allow to remove attacks between arguments accordingly, thus, modifying the set of extensions obtained.

Application Argumentation has been used for *handling inconsistency* in knowledge bases, merging several knowledge bases and making decisions under uncertainty. In the framework of the INRA GloFood Pack4Fresh project, it was decided that a *decision support system* based on argumentation could be used to select the best strawberries packaging according to possibly conflicting requirements provided by multiples stakeholders. In order to do this, we analyze how to instantiate argumentation frameworks and study how preferences behave on such frameworks. For instance, one can discuss the pros and cons of using shock mount, vacuum or a small wooden strawberries packaging (or a mix of them). Small wooden packaging may be reusable but may hurt the content in the event of shocks. Food that comes in vacuum packaging will also stay fresh in the freezer longer than food wrapped in other types of packages but the plastic layer may not be good for the environment. Shock mount packaging are good for protecting the contents from shock and vibration as well as humidity, dust, and moisture but also increase the final price.

Arguments for Agronomy In [26, 27], the authors introduced an argumentation system capable of selecting generic packaging types (the language used is less expressive than $Datalog^{\pm}$). In [15], the authors coupled reverse-engineering and argumentation in the bread making industry in order to solve a new type of problem in which the demand (and not the supply) sets the specifications of desired products and it is up to the supply to adapt and find its ways to respond. One of the main research question is what happens when we take into account stakeholders' preferences. State of the art covers this using different languages not clearly put in context. The aim of this study is to clarify all this. Moreover, it should be noted that although the software introduced in the European project *EcoBioCap* [27, 26] is close to what we need, we had to add a preferences module in order to take account preferences on choices.

Thesis Structure This study investigates the state of art with respect to research in the field of Artificial Intelligence, more precisely in handling preferences in logic-based argumentation frameworks. The research question of this study revolves around answering two questions: how to instantiate an argumentation framework with existential rules and how this impacts the role of preferences on decision making over such systems?

In the first chapter of this paper, we will recall the theoretical background behind this study. Namely, we recall the basics under the family of logical languages $Datalog^{\pm}$ and explain briefly how we can handle inconsistencies. We recall that an argumentation system is composed of a set of arguments and attacks between them. Then, to reach a conclusion, we use particular acceptability semantics defined in [18] to calculate a set of arguments called *extensions* that satisfy these semantics.

In the second chapter, we will explain the two state of the art possibilities to instantiate a logical argumentation framework with $Datalog^{\pm}$. The first method to instantiate was proposed in [17, 16] and the second way is to use a logically instantiated argumentation system such as $ASPIC^{+}$framework proposed in [11, 22, 23, 29, 20]. In this study we consider the latest version of $ASPIC^{+}$[20]. It is important to note that in order to instantiate $ASPIC^{+}$, some requirements are necessary. Next, we will show that there is an equivalence between sets of arguments produced by the two frameworks.

The third chapter is about preferences in the different logical argumentation frameworks. The first method introduced in [16] represents preferences as a relation on facts in order to take into account the level of significance of the knowledge expressed by the various ontological knowledge sources. Using this method, one can extract different subset of extensions (locally optimal, Pareto optimal and globally optimal extensions) of the set obtained under a particular semantics. The second method (used in $ASPIC^{+}$) sees preferences as a binary ordering on arguments and removes attacks between arguments accordingly, thus, modifying the set of extensions obtained. We will show that there is an equivalence between Pareto optimal extensions and $ASPIC^{+}$stable extensions with a specific preference semantics. One of our goals is to implement the first method in $ASPIC^{+}$because it does not alter the set of extensions unlike $ASPIC^{+}$'s default method does.

In the last chapter, we explain a use-case and explicitly show the results obtained and how they validate the previous equivalence. Finally, we show the different functionalities of the existing argumentation-based software introduced in the framework of the European project *EcoBioCap* [27, 26] and point out what was modified to satisfy the needs of this internship.

2

Logic-Based Argumentation Frameworks

In this chapter, we first recall the theoretical background needed to comprehend this study, then we focus on structured argumentation (i.e. argumentation frameworks instantiated with a logical language) and explain two recent methods of instantiating using $Datalog^{\pm}$. Please note that throughout this chapter, we will use two argumentation frameworks. In order to differentiate the two frameworks, we will denote the first instantiation associated with $Datalog^{\pm}$ by $\mathcal{AS}^{M}_{\mathcal{K}}$ and the second new instantiation by $\mathcal{AS}^{A}_{\mathcal{K}}$ with $\mathcal{K}$ being a knowledge base. All proofs of this chapter are available in Appendix D.

2.1 Theoretical Background

Ontologies are an important notion in knowledge representation as they bring information on the domain we are using. For example, imagine that the data is composed of different kind of strawberries, then the ontology will provide the vocabulary needed to describe each strawberry. The first part of this section deals with the basic bricks of knowledge representation in $Datalog^{\pm}$ (existential rules) and the second part describes the most popular semantics proposed by Dung in his seminal paper [18].

2.1.1 Existential Rules And Repairs

There are two major approaches to represent an ontology: Description Logics (such as $\mathcal{EL}$ [7] and DL-Lite families [13]) and rule-based languages (such as $Datalog^{\pm}$ language [12], a generalization of Datalog that allows for existentially quantified variables in rules heads). Despite $Datalog^{\pm}$ undecidability when answering conjunctive queries, different decidable fragments are studied in the literature [10]. These fragments generalize the aforementioned Description Logics families and overcome their limitations by allowing any predicate arity as well as cyclic structures. Here we use the general rule-based setting knowledge representation language $Datalog^{\pm}$, i.e, *the positive existential* conjunctive fragment of first-order logic FOL [14, 9]. Its language $\mathcal{L}$ is composed of formulas built with the usual quantifier $(\exists, \forall)$ and *only* the connectors implication $(\rightarrow)$ and conjunction $(\wedge)$. The following definitions are classic in the literature but needed to properly understand the difference between the two instantiations in the state of the art.

Vocabulary We consider first-order vocabularies with constants but no other function symbol. A vocabulary is a pair $\mathcal{V} = (\mathcal{P}, \mathcal{C})$, where $\mathcal{P}$ is a finite set of predicates and $\mathcal{C}$ is a possibly infinite set of constants. A term t over $\mathcal{V}$ is a constant or a variable, different constants represent different values (unique name assumption). Please note that we will use names of strawberries (Allstar, Surecrop, Earliglow, etc.) as constants in the examples if not stated otherwise and lowercase letters (x, y, etc.) for variables.

Atomic formulae An **atomic formula** (or atom) over $\mathcal{V}$ is of the form $p(t_1, ..., t_n)$ where $p \in \mathcal{P}$ is an n-ary predicate, and $t_1, ..., t_n$ are terms. A **ground atom** is an atom with no variables (e.g. $strawberry(Allstar)$). A conjunction of atoms is called a **conjunct**. A conjunction of **ground atoms** is called a **ground conjunct**. By convention a ground atom is a ground conjunct. A variable in a formula is free if it is not in the scope of any quantifier. A formula is **closed** if it has no free variables (also known as **sentence**).

Example 1 (Atoms and conjuncts) *Let us consider the set of constants* $\mathcal{C} = \{Allstar, Surecrop\}$, $\mathcal{P} = \{sweetFlavor, diseaseResist, sweeter\}$ *and a countably infinite set of variables* $\mathcal{X} = \{x_1, x_2, \ldots\}$. *Then,* $sweeter(Allstar, x_1)$ *is an atom,* $diseaseResist(Allstar)$ *is a ground atom,* $sweeter(Allstar, x_1) \wedge diseaseResist(Allstar)$ *is a conjunct and* $diseaseResist(Allstar) \wedge sweeter(Surecrop, Allstar) \wedge sweetFlavor(Allstar)$ *is a ground conjunct.*

Facts Classically, a fact is a ground atom. The notion was extended in [9], so that a fact may contain existentially quantified variables and not only constants. Thus, a **fact** on $\mathcal{V}$ is the existential closure of a conjunction of atoms over $\mathcal{V}$. For instance, $F = \exists x_1(sweetFlavor(Allstar) \wedge diseaseResist(x_1) \wedge sweeter(Allstar, x_1))$ is an example of a fact where x_1 is an existentially quantified variable. We may omit quantifiers in facts as there is no ambiguity (they are all existentially quantified). As one may notice here, the existential variable permit to represent unknown values which is an interesting property in this language. We denote by *terms*(F) (resp. *vars*(F)) the set of terms (resp. variables) that occur in F. We exclude duplicate atoms in facts, which allows to see a fact as a set of atoms. For instance, the fact $F = \exists x \exists y(redInside(x) \wedge sweeter(Allstar, y))$ can be seen as $\{sweeter(Allstar, y), redInside(x)\}$ where *vars*$(F) = \{x, y\}$ and *terms*$(F) = \{Allstar\}$. From now on we may use the set notation and the logical notation interchangeably.

Rules and negative constraints We denote by $\vec{x}$ a vector of variables. An *existential rule* (or simply a rule) is a closed formula of the form $R = \forall \vec{x} \forall \vec{y}(B \rightarrow \exists \vec{z} H)$, where B and H are conjuncts, with *vars*$(B) = \vec{x} \cup \vec{y}$, and *vars*$(H) = \vec{x} \cup \vec{z}$. The variables $\vec{z}$ are called the existential variables of the rule R. B and H are respectively called the *body* and the *head* of R. We denote them respectively $body(R)$ for B and $head(R)$ for H. We may sometimes omit quantifiers and write $R = B \rightarrow H$. As an example of rules we have $R = \forall x \forall y\ smallerSize(y, x) \rightarrow \exists z\ biggerSize(z, y)$. A *negative constraint* (or simply a constraint) is a rule of the form $N = \forall \vec{x}(B \rightarrow \bot)$. For instance, $\forall x\ redSkin(x) \wedge greenSkin(x) \rightarrow \bot$ is a negative constraint that says that x cannot have $redSkin$ and $greenSkin$.

Substitution and homomorphism Given a set of variables $\mathcal{X}$ and a set of terms $\mathcal{T}$, a **substitution** σ of $\mathcal{X}$ by $\mathcal{T}$ (notation $\sigma : \mathcal{X} \rightarrow \mathcal{T}$) is a mapping from $\mathcal{X}$ to $\mathcal{T}$. Given a fact F, $\sigma(F)$

denotes the fact obtained from F by replacing each occurrence of $x \in X \cap vars(F)$ by $\sigma(x)$. A **homomorphism** from a fact F to a fact F' is a substitution σ of $vars(F)$ by (a subset of) $terms(F')$ such that $\sigma(F) \subseteq F'$ [9].

Example 2 (Homomorphism) *Let* $F = \{sweeter(Surecrop, x)\}$ *and* $F' = \{sweeter(Surecrop, Allstar), sweetFlavor(Allstar)\}$ *where* $vars(F) = \{x\}$ *and* $terms(F') = \{Allstar, Surecrop\}$. *We have two possible substitutions* $\sigma_1 = \{(x, Surecrop)\}$ *and* $\sigma_2 = \{(x, Allstar)\}$ *where* x *is substituted by* $Surecrop$ *in* σ_1 *and by* $Allstar$ *in* σ_2. *When we apply* σ_1 *(resp.* σ_2*) on* F *we get* $\sigma_1(F) = \{sweeter(Surecrop, Surecrop)\}$ *(resp.* $\sigma_2(F) = \{sweeter(Surecrop, Allstar)\}$*). It is clear that the substitution* σ_2 *is a homomorphism from* F *to* F' *(unlike* σ_1*) because* $\sigma_2(F) \subseteq F'$ *such that* $\sigma_2(F) = \{sweeter(Surecrop, Allstar)\}$.

Rules Application A rule $R = B \rightarrow H$ is **applicable** to a fact F if there is a homomorphism σ from B to F. The *application of* R *to* F *w.r.t.* σ produces a fact $\alpha(F, R, \sigma) = F \cup \sigma(safe(H))$, where *safe*$(H)$ is obtained from H by replacing existential variables with fresh variables (not used variables). $\alpha(F, R, \sigma)$ is said to be an **immediate derivation** from F. For instance, let $R = redder(x, y) \rightarrow moreAttractive(x, y)$ and $F = \{redder(Earliglow, Honeoye), redSkin(Earliglow)\}$, R is applicable to F because there is a homomorphism from $\{redder(x, y)\}$ to $\{redder(Earliglow, Honeoye), redSkin(Earliglow)\}$ that substitutes x by $Earliglow$ and y by $Honeoye$. The immediate derivation from F is the fact $F' = \{redder(Earliglow, Honeoye), redSkin(Earliglow)\} \cup \{moreAttractive(Earliglow, Honeoye)\}$.

Derivation and closure Let F be a fact and $\mathcal{R}$ be a set of rules. A fact F' is called an **$\mathcal{R}$-derivation** of F if there is a finite sequence (called the **derivation sequence**) $(F_0 = F, ..., F_n = F')$ such that for all $0 \leq i < n$ there is a rule R which is applicable to F_i and F_{i+1} is an immediate derivation from F_i. Given a fact F and a set of rules $\mathcal{R}$, the **chase** (or saturation) procedure starts from F and performs rule applications in a breadth-first manner. The chase computes the **closure** of F, i.e. $CL_{\mathcal{R}}(F)$, which is the smallest set that contains F and that is closed under R-derivation, i.e. for every $\mathcal{R}$-derivation F' of F we have $F' \in CL_{\mathcal{R}}(F)$. Given a chase variant C [8], we call C-finite the class of set of rules $\mathcal{R}$, such that the C-chase halts on any fact F, consequently produces a finite $CL_{\mathcal{R}}(F)$. We limit the work of this paper to these kind of classes.

Entailment Let F and F' be two facts. $F \models F'$ if and only if there is a homomorphism from F' to F. For instance $\{betterJam(Sparkle, x), vigorous(Sparkle)\} \models betterJam(Sparkle, Allstar)$. Given two facts F and F' and a set of rules $\mathcal{R}$ we say $F, \mathcal{R} \models F'$ if and only if $CL_{\mathcal{R}}(F) \models F'$ where $\models$ is the classical first-order entailment [21].

Example 3 ($\mathcal{R}$-derivation) *For readability of this example only, we will use simple notation for predicates names. Let* $F = \{q(A, B), r(D), p(x_1, C)\}$ *and* $\mathcal{R} = \{R_1, R_2\}$ *such that* $R_1 = q(x_1, y_1) \wedge r(z_1) \rightarrow d(x_1, z_1)$ *and* $R_2 = p(x_2, y_2) \wedge r(z_2) \rightarrow m(z_2, x_2)$. *The following is a derivation sequence:* $\langle F_0, F_1, F_2 \rangle$ *where* $F_0 = F$, $F_1 = \{q(A, B), r(D), d(A, D), p(x_1, C)\}$ *and* $F_2 = F_1 \cup \{m(D, x_1)\}$. *We get* F_1 *by applying* $\mathcal{R}_1$ *on* F *then we get* F_2 *by applying* R_2 *on* F_1. *We say* F_2 *is an* $\mathcal{R}$*-derivation of* F. *The closure of* F *is* $CL_{\mathcal{R}}(F) = F \cup \{d(A, D), m(D, x_1)\}$.

Knowledge base and inconsistency Let us denote by $\mathcal{L}$ the language described so far, A knowledge base $\mathcal{K}$ is a finite subset of $\mathcal{L}$. Precisely, $\mathcal{K}$ is a tuple $(\mathcal{F}, \mathcal{R}, \mathcal{N})$ of a finite set of facts $\mathcal{F}$, rules $\mathcal{R}$ and constrains $\mathcal{N}$. Saying that $\mathcal{K} \models F$ means $CL_{\mathcal{R}}(\mathcal{F}) \models F$. We say a set of facts $\mathcal{F}$ is $\mathcal{R}$-inconsistent with respect to a set of constraints $\mathcal{N}$ and rules $\mathcal{R}$ if and only if there exists $N \in \mathcal{N}$ such that $CL_{\mathcal{R}}(\mathcal{F}) \models body(N)$, otherwise $\mathcal{F}$ is $\mathcal{R}$-consistent. A knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ is said to be inconsistent with respect to $\mathcal{R}$ and $\mathcal{N}$ (inconsistent for short) if $\mathcal{F}$ is $\mathcal{R}$-inconsistent. We may use the notation $CL_{\mathcal{R}}(\mathcal{F}) \models \bot$ to mean the same thing.

Example 4 (Knowledge base) *The following is an example of a knowledge base.*

- $\mathcal{F} = \{strawberry(Surecrop), deepRedSkin(Sparkle)\}$
- $\mathcal{R} = \{\forall x_1(strawberry(x_1) \rightarrow fruit(x_1)),$
  $\forall x_1 \forall x_2(deepRedSkin(x_1) \wedge redSkin(x_2) \rightarrow redder(x_1, x_2))\}$
- $\mathcal{N} = \{\forall x_1(deepRedSkin(x_1) \wedge redSkin(x_1) \rightarrow \bot)\}$

Repairs It is important to be able to handle inconsistencies as everything can be inferred by an inconsistent knowledge base. In the area of inconsistent ontological knowledge base query answering, we usually check what can be inferred from an inconsistent ontology. We usually begin by calculating all maximal consistent subsets of $\mathcal{K}$ called *repairs*. Given a knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$, we call by $Repairs(\mathcal{K})$ the set of all repairs defined as:

$$Repairs(\mathcal{K}) = \{\mathcal{F}' \subseteq \mathcal{F} | \mathcal{F}' \text{ is maximal for } \subseteq \text{ and } \mathcal{R}\text{-consistent}\}$$

Example 5 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base such that we have $\mathcal{F} = \{strawberry(Allstar), basketball(Allstar), Thing(Allstar)\}$, $\mathcal{R} = \{\forall x(strawberry(x) \rightarrow fruit(x))\}$ and $\mathcal{N} = \{\forall x(strawberry(x) \wedge basketball(x) \rightarrow \bot)\}$. It is obvious that this knowledge base is inconsistent and $Repairs(\mathcal{K}) = \{\{strawberry(Allstar), Thing(Allstar)\}, \{basketball(Allstar), Thing(Allstar)\}\}$.*

Different *inconsistent tolerant semantics* on repairs have been studied in [17] and detailed in Appendix A to handle inconsistencies. Moreover, *Default Logic* defined in [24] and detailed in Appendix B and *Argumentation* are also other methods capable of repairing inconsistent knowledge bases. All three methods produce maximal consistent subsets of the knowledge base called extensions (in argumentation or default logic) or repairs. In this study, we choose to focus on argumentation because it provides a natural way to interact with users (via dialogues as investigated by [6]) that is useful in practical applications. Next chapter shows that this choice is of no cost with respect to expressivity.

2.1.2 Dung's Abstract Argumentation

An *abstract argumentation framework* is a pair $\mathcal{AS} = (\mathcal{A}, \mathcal{C})$, where $\mathcal{A}$ is a set of arguments and $\mathcal{C} \subseteq \mathcal{A} \times \mathcal{A}$ is a binary attack relation on them. Note that in an abstract argumentation framework, we do not know how the structure of the arguments nor how the attack is constructed. A semantics for an argumentation framework returns a set of arguments called extensions, which are internally conflict-free and defend themselves from the other arguments. In the following definition, we define the most popular semantics proposed by Dung in [18].

Definition 1 ***[cf [18]].*** *Given an argumentation framework $\mathcal{AS} = (\mathcal{A}, \mathcal{C})$. We say that an argument $a \in \mathcal{A}$ is* acceptable *w.r.t a set of arguments $\varepsilon \subseteq \mathcal{A}$ iff $\forall b \in \mathcal{A}$ such that $(b, a) \in \mathcal{C}, \exists c \in \varepsilon$ such that $(c, b) \in \mathcal{C}$. Moreover, an extension can follow different semantics:*

- *ε is* conflict-free *iff $\nexists a, b \in \varepsilon$ such that $(a, b) \in \mathcal{C}$.*
- *ε is* admissible *iff ε is conflict-free and all arguments of ε are acceptable w.r.t ε.*
- *ε is preferred iff it is maximal (for set inclusion) and admissible.*
- *ε is* stable *iff it is conflict-free and $\forall a \in \mathcal{A} \backslash \varepsilon, \exists b \in \varepsilon$ such that $(b, a) \in \mathcal{C}$.*
- *ε is* complete *iff it contains all arguments that are acceptable w.r.t ε.*
- *ε is* grounded *iff it is minimal (for set inclusion) and complete.*

Please find some examples of extensions under different semantics in Appendix C. In order to apply the theory, an instantiation of Dung's abstract argumentation frameworks is needed. An *instantiation* starts with a knowledge base and constructs the arguments and attacks. Then, given the set of arguments and the attack relation, extensions and conclusions are produced. The next section focus on giving two possible instantiations for $Datalog^{\pm}$.

2.2 Two Possible Instantiations

2.2.1 An Existing Instantiation For $Datalog^{\pm}$

The following definition specifies the structure of an argument in the first instantiation. An argument in this instantiation is composed of a derivation sequence $(\mathcal{F}_0, \mathcal{F}_1, \ldots, \mathcal{F}_{n-1})$ where $\mathcal{F}_0$ is a subset of $\mathcal{F}$ and an element $\mathcal{F}_n$ s.t. $\mathcal{F}_{n-1} \models \mathcal{F}_n$.

Definition 2 (Arguments structure) ***[cf [17, 16]].*** *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, an argument a as described in [16, 17] is a tuple $a = (\mathcal{F}_0, \mathcal{F}_1, \ldots, \mathcal{F}_n)$ where:*

- *$\mathcal{F}_0 \subseteq \mathcal{F}$ is $\mathcal{R}$-consistent and $(\mathcal{F}_0, \mathcal{F}_1, \ldots, \mathcal{F}_{n-1})$ is a derivation sequence.*
- *$\mathcal{F}_n$ is an atom, a conjunction of atoms, the existential closure of an atom or the existential closure of a conjunction of atoms such that $\mathcal{F}_{n-1} \models \mathcal{F}_n$.*

We denote by $Supp(a) = \mathcal{F}_0$ the support of an argument, $Conc(a) = \mathcal{F}_n$ its conclusion and $\forall X \subseteq \mathcal{F}, Arg(X)$ is the set of all arguments a such that $Supp(a) \subseteq X$.

Example 6 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base where $\mathcal{F} = \{strawberry(Allstar), basketball(Allstar)\}$, $\mathcal{R} = \{\forall x\,(strawberry(x) \rightarrow fruit(x))\}$ and $\mathcal{N} = \{\forall x, (strawberry(x) \wedge basketball(x) \rightarrow \bot)\}$. One argument can be $a_1 = (\{strawberry(Allstar)\}, \{strawberry(Allstar), fruit(Allstar)\}, fruit(Allstar))$ but also $a_2 = (\{basketball(Allstar)\}, \exists x(basketball(x)))$ because $basketball(Allstar) \models \exists x(basketball(x))$.*

The next step is to define the attack relation between arguments. Please note that in [17], the authors considered that in an inconsistent knowledge base, only the set of facts is considered as inconsistent whereas the set of rules is considered as consistent. Thus, the attack relation is an *undermining*, roughly speaking, all the inconsistency "comes from the facts". Therefore, there is an attack from argument a to argument b iff the conclusion of a and an element of the support of b are $\mathcal{R}$-inconsistent. This is an important step because it enables us to calculate sets of accepted arguments via semantics. Please note that in [3], the author gives advices as how to define an appropriate attack relation.

Definition 3 (Attacks) ***[cf [17, 16]].*** *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base and let a and b be two arguments. We say that an argument a attacks an argument b denoted by $(a, b) \in Att$ iff $\exists \varphi \in Supp(b)$ such that $\{Conc(a), \varphi\}$ is $\mathcal{R}$-inconsistent.*

This attack relation is not symmetric as expressed in Example 7. Moreover, it have been showed that symmetric attack relations violate some desirable properties [3]. Using Definition 2, the logical argumentation framework associated with a $Datalog^{\pm}$ knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ is $\mathcal{AS}_{\mathcal{K}}^{M} = (\mathcal{A}, \mathcal{C})$ with $\mathcal{A} = Arg(\mathcal{F})$ and $\mathcal{C}$ defined in definition 3.

Example 7 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base where $\mathcal{F} = \{\ strawberry(Chandler)$, $greenSkin(Chandler)$, $redSkin(Chandler)\}$, $\mathcal{R} = \emptyset$ and $\mathcal{N} = \{\forall x\ (\ strawberry(x) \wedge greenSkin(x) \wedge redSkin(x) \rightarrow \bot\ )\}$. We get that the argument $a_1 = (\{strawberry(Chandler), greenSkin(Chandler)\}, strawberry(Chandler) \wedge greenSkin(Chandler))$ attacks argument $a_2 = (\{redSkin(Chandler)\}, redSkin(Chandler))$ but a_2 does not attak a_1 because there is no $\varphi \in Supp(a_1)$ s.t. $\{Conc(a_2), \varphi\}$ is $\mathcal{R}$-inconsistent.*

As said before, an extension is a set of arguments. Let $\mathcal{AS}$ be an argumentation system, we denote by $Ext_x(\mathcal{AS})$ the set of its extensions with respect to semantics x. We also call the *base* of an extension the set of facts that the arguments of the extension are based on.

Definition 4 (Base) ***[cf [17, 16]].*** *Let ε be an extension, the base of ε is $Base(\varepsilon) = \bigcup_{a \in \varepsilon} Supp(a)$.*

We will now introduce the second new instantiation for $Datalog^{\pm}$ using $ASPIC^{+}$.

2.2.2 A New Instantiation using $ASPIC^{+}$For $Datalog^{\pm}$

$ASPIC^{+}$[20] is a framework for obtaining logical based argumentation system using any logical language $\mathcal{L}$. It is meant to generate an abstract argumentation framework and was created because abstract argumentation does not specify the structure of arguments and the nature of attacks. $ASPIC^{+}$is meant to provide guidance to those aspects without losing a large range of instantiating logics. According to [20], in order to use $ASPIC^{+}$, we need to choose a logical language $\mathcal{L}$ closed under negation ($\neg$), to provide a set of rules $\mathcal{R} = \mathcal{R}_d \cup \mathcal{R}_s$ composed of defeasible rules and strict rules with $\mathcal{R}_d \cap \mathcal{R}_s = \emptyset$, and to specify a contrariness function $cf : \mathcal{L} \rightarrow 2^{\mathcal{L}}$ and a partial naming function $n : \mathcal{R}_d \rightarrow \mathcal{L}$ that associates a well-formed formula of $\mathcal{L}$ to a defeasible rule. The function n will not be used in this instantiation. In $ASPIC^{+}$, an argumentation system is a triple $\mathcal{AS}_{\mathcal{K}}^{A} = (\mathcal{L}, \mathcal{R}, n)$ with $\mathcal{K} \subseteq \mathcal{L}$ being a knowledge base consisting of two disjoint subsets $\mathcal{K}_n$ (the axioms) and $\mathcal{K}_p$ (the ordinary premises).

Let us now instantiate $ASPIC^+$. We will define $\mathcal{L}$ as $Datalog^{\pm}$, rules in Definition 5 and the contrariness function in Definition 6. Please note that Definition 7 and 10 are new w.r.t state of art [20, 27]. A rule in $ASPIC^+$has the same form as an existential rule except for the fact that it can either be strict ($\rightarrow$) or defeasible ($\Rightarrow$).

Definition 5 (Rules in $ASPIC^+$) *Strict rules (resp. defeasible rules) are of the form $\forall\vec{x}\forall\vec{y}(B \rightarrow \exists\vec{z}H)$ (resp. $\forall\vec{x}\forall\vec{y}(B \Rightarrow \exists\vec{z}H)$) with B, the body and H, the head are atoms or conjunction of atoms with $\text{vars}(B) = \vec{x} \cup \vec{y}$, and $\text{vars}(H) = \vec{x} \cup \vec{z}$..*

The next definition explicitly states the difference between the contrary and the contradictory of an element. The main difference is that the contrary is not symmetric whereas the contradictory is.

Definition 6 ***[cf [1, 20]].*** *cf is a function from $\mathcal{L}$ to $2^{\mathcal{L}}$ such that:*

- *φ is the contrary of ψ if $\varphi \in cf(\psi), \psi \notin cf(\varphi)$*
- *φ is the contradictory of ψ if $\varphi \in cf(\psi), \psi \in cf(\varphi)$*
- *Each $\varphi \in \mathcal{L}$ has at least one contradictory.*

It is worth noting that the idea that we want to capture as defined in [1] is that x is the contrary of y iff they cannot be both true but they can be both false. They are contradictory if the truth of one implies the falsity of the other and vice versa. Please note that Definition 6 does not always satisfy that idea.

Example 8 ***[cf [1]]*** *Let $\mathcal{L} = \{\geq, >, =, \neq, <, \leq\}$ be a language. Then we have $cf(>) = \{<, \leq, =\}$ and $cf(<) = \{>, \geq, =\}$. We have that $>$ is the contrary of $<$ but according to definition 6, they are contradictory while this is not the case.*

We define the contrariness function to instantiate $ASPIC^+$with $Datalog^{\pm}$ as the language. The contrariness function cf is a key element to define the attacks in $ASPIC^+$and returns the set of contraries of an element. We say that an atom or a conjunction of atoms belongs to $cf(a)$ where a is an element of the language if there exists ψ such that $a \models \psi$ and $\{b, \psi\}$ is $\mathcal{R}$-inconsistent.

Definition 7 (new definition) *Let $a \in \mathcal{L}$ and b be an atom or a conjunction of atoms. $b \in cf(a)$ iff $\exists\psi$ an atom such that $a \models \psi$ and $\{b, \psi\}$ is $\mathcal{R}$-inconsistent.*

Example 9 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base where $\mathcal{F} = \{strawberry(Chandler)$, $greenSkin(Chandler)$, $redSkin(Chandler)\}$, $\mathcal{R}=\emptyset$ and $\mathcal{N}=\{\forall x\ (strawberry(x) \wedge greenSkin(x) \wedge redSkin(x) \rightarrow \bot)\}$. We get that:*

- $cf(strawberry(Chandler)) = \{strawberry(Chandler) \wedge greenSkin(Chandler) \wedge redSkin(Chandler), greenSkin(Chandler) \wedge redSkin(Chandler)\}$.
- $cf(greenSkin(Chandler) \wedge redSkin(Chandler)) = \{strawberry(Chandler) \wedge greenSkin(Chandler) \wedge redSkin(Chandler)$, $strawberry(Chandler) \wedge greenSkin(Chandler)$, $strawberry(Chandler) \wedge redSkin(Chandler)\}$.

Here we recall that an $ASPIC^+$argument can be built from axioms and ordinary premises or from rules and other arguments. The arguments are built once $\mathcal{R}_d, \mathcal{R}_s, cf$ and $\mathcal{K}$ are known.

Definition 8 (modified definition) ***[cf [20, 27, 1]].*** *Arguments in $ASPIC^+$can be in two forms:*

- $\emptyset \rightarrow c$ *(resp.* $\emptyset \Rightarrow c$*) with* $c \in \mathcal{K}_n$ *(resp.* $c \in \mathcal{K}_p$ *or* $\emptyset \Rightarrow c \in \mathcal{R}_d$*) such that* $Prem(A) = \{c\}, Conc(A) = c,$ ***Sub***$(A) = \{A\}$ *with* $Prem$ *returns the premises of* A *and* $Conc$ *returns its conclusion.*
  $DefRules(A) = \emptyset$.

- $A_1, ..., A_m \rightarrow c$ *(resp.* $A_1, ..., A_m \Rightarrow c$*), such that there exists a strict (resp. defeasible) rule* $r = B \rightarrow H$ *(resp.* $r = B \Rightarrow H$*) and a homomorphism* σ *from* B *to* $X = Conc(A_1) \wedge Conc(A_2) \wedge \cdots \wedge Conc(A_m)$.
  $Prem(A) = Prem(A_1) \cup ... \cup Prem(A_m)$,
  $Conc(A) = c = \alpha(X, r, \sigma)$,
  Sub$(A) =$ ***Sub***$(A_1) \cup ... \cup$ ***Sub***$(A_m) \cup \{A\}$,
  $TopRule(A) =$ *rule* $r = B \rightarrow H$ *(resp.* $r = B \Rightarrow H$*), such that there exists an homomorphism* σ *from* B *to* X.
  $DefRules(A) = DefRules(A_1) \cup \cdots \cup DefRules(A_m)$ *(resp.* $DefRules(A) = DefRules(A_1) \cup \cdots \cup DefRules(A_m) \cup \{TopRule(A)\}$*).*

Attacks in $ASPIC^+$are particular in the sense that they are based on three notions (undercutting, undermining and rebutting). Each of those notions are useful as they capture different aspects of conflicts. In short, arguments can be attacked on a conclusion of a defeasible inference (rebutting attack), on a defeasible inference step itself (undercutting attack), or on an ordinary premise (undermining attack).

Definition 9 ***[cf [20]].*** *Let* a *and* b *be arguments, we say that* a *attacks* b *iff* a *undercuts, undermines or rebuts* b*, where:*

- a *undercuts argument* b *(on* b'*) iff* $Conc(a) \in cf(n(r))$ *for some* $b' \in Sub(b)$ *such that* b'*'s top rule* r *is defeasible.*

- a *rebuts argument* b *(on* b'*) iff* $Conc(a) \in cf(\psi)$ *for some* $b' \in Sub(b)$ *of the form* $b'_0, \ldots, b'_n \Rightarrow \psi$.

- a *undermines* b *(on* ψ*) iff* $Conc(a) \in cf(\psi)$ *for an ordinary premise* ψ *of* b.

The next definition explains how we can match a knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ with an $ASPIC^+$instantiation by matching consistent subsets of $\mathcal{F}$ to the corresponding ordinary premises and rules of $\mathcal{R}$ to the corresponding strict rules.

Definition 10 (new definition) *We denote by* $\mathcal{S}$ *the set of all possible inconsistent knowledge bases of the form* $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ *and* $\mathcal{G}$ *the set of all* $ASPIC^+$*instantiations using* $Datalog^{\pm}$ *language. The mapping* $\tau : S \rightarrow \mathcal{G}$ *is defined as follows:*

(a) The mapping τ *associates every* $\mathcal{R}$*-consistent subset* $F_i \subseteq \mathcal{F}$ *to its defeasible rule* $\emptyset \Rightarrow conjunct(F_i)$ *where* $conjunct(F_i)$ *denotes the conjunction of facts contained in* F_i.

(b) The mapping τ associates every rules $r_i \in \mathcal{R}$ to the same rule $r_i \in \mathcal{R}_s$.

We will consider that if $\emptyset \Rightarrow c$, then c is an ordinary premise ($c \in \mathcal{K}_p$).

Example 10 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base such that $\mathcal{F}$=$\{strawberry(Allstar)$, $basketball(Allstar)\}$, $\mathcal{R}$=$\{\forall x\,(strawberry(x) \rightarrow fruit(x)), \forall x(basketball(x) \rightarrow sport(x))\}$ and $\mathcal{N} = \{\forall x(fruit(x) \wedge sport(x) \rightarrow \bot), \forall x(fruit(x) \wedge basketball(x) \rightarrow \bot)\}$. We call $\mathcal{AS}^A_{\mathcal{K}} = (\mathcal{L}, \mathcal{R}', n)$ the corresponding $ASPIC^+$ argumentation system with $\mathcal{R}' = \{\emptyset \Rightarrow strawberry(Allstar)$, $\emptyset \Rightarrow basketball(Allstar)\} \cup \mathcal{R}$. This argumentation framework contains four arguments:*

- $A_1 : \emptyset \Rightarrow strawberry(Allstar)$
- $A_2 : A_1 \rightarrow fruit(Allstar)$
- $B_1 : \emptyset \Rightarrow basketball(Allstar)$
- $B_2 : B_1 \rightarrow sport(Allstar)$

If we use the previous definition 7 of the contrariness function, we get that $fruit(Allstar) \in cf(sport(Allstar))$, $fruit(Allstar) \in cf(basketball(Allstar))$, $sport(Allstar) \in cf(fruit(Allstar))$ and $basketball(Allstar) \in cf(fruit(Allstar))$. We can conclude that A_2 undermines B_2 on $basketball(Allstar)$.

2.3 Semantics Equivalence Between The Two Instantiations

In this section, we want to prove some equivalence results on both argumentation frameworks. Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, we recall that $\mathcal{AS}^M_{\mathcal{K}}$ denotes the first instantiated logical argumentation framework and $\mathcal{AS}^A_{\mathcal{K}}$ the second new $ASPIC^+$ argumentation framework constructed from $\mathcal{K}$ using the mapping of Definition 10. Please note that attacks in $\mathcal{AS}^A_{\mathcal{K}}$ are composed only of undermining because we only have simple defeasible rules of the form $\emptyset \Rightarrow c$.

The following lemma shows that stable extensions are closed under sub-arguments in the $Datalog^{\pm}$ instantiation of $ASPIC^+$.

Lemma 2.3.1 *Let ε be an $ASPIC^+$ stable extension and $A \in \varepsilon$ an argument contained in ε. Then $Sub(A) \subseteq \varepsilon$.*

Notation Let $c = \alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_n$ be a conjunction of facts. $Elimination(c) = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ is the set resulting from eliminating the conjunctions of c. Let S be a set of facts. We denote by $\mathcal{P}(S)$ the superset of S which corresponds to all subsets of S.

We can now define the set of arguments constructed on a consistent set of facts.

Definition 11 (new definition) *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base and $\mathcal{AS}^A_{\mathcal{K}}$ be the corresponding $ASPIC^+$ instantiation and $S \subseteq \mathcal{F}$ a $\mathcal{R}$-consistent subset of $\mathcal{F}$. We denote by $Arg^A(S)$ the set of arguments such that their premises are contained in S.*

$$Arg^A(S) = \{ASPIC^+ \textit{ argument } a \mid \bigcup_{c \in Prem(a)} Elimination(c) \subseteq \mathcal{P}(S)\}$$

The main result shows that the set of stable extensions coincides with the set of preferred extensions and it is obtained from the arguments built on repairs.

Theorem 2.3.2 (new theorem) *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{AS}^A_{\mathcal{K}}$ be the corresponding $ASPIC^+$ instantiation and $\sigma \in \{preferred, stable\}$. Then:*

$$\{Arg^A(R) | R \in Repair(\mathcal{K})\} = Ext_\sigma(\mathcal{AS}^A_{\mathcal{K}})$$

In [17], the authors showed the same result equivalence between the set of stable and preferred extensions of $Ext_\sigma(\mathcal{AS}^M_{\mathcal{K}})$.

Theorem 2.3.3 ***[cf [17]]*** *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{AS}^M_{\mathcal{K}}$ be the corresponding logical argumentation framework instantiation and $\sigma \in \{preferred, stable\}$. Then:*

$$\{Arg(R) | R \in Repair(\mathcal{K})\} = Ext_\sigma(\mathcal{AS}^M_{\mathcal{K}})$$

where Arg is defined in definition 2 and is different than Arg^A.

Using Theorem 2.3.2 and Theorem 2.3.3 we can conclude that there is the same number of preferred/stable extensions in the two frameworks and that for each stable/preferred extension of one framework, there is a corresponding stable/preferred extension in the other and vice versa.

Theorem 2.3.4 (new theorem) *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{AS}^M_{\mathcal{K}}$ and $\mathcal{AS}^A_{\mathcal{K}}$ be the two argumentation framework instantiations. Then if $\sigma \in \{preferred, stable\}$, $|Ext_\sigma(\mathcal{AS}^M_{\mathcal{K}})|$ = $|Ext_\sigma(\mathcal{AS}^A_{\mathcal{K}})|$ and for each extension under semantics $\sigma, \varepsilon \in Ext_\sigma(\mathcal{AS}^M_{\mathcal{K}})$, there is a corresponding extension $\varepsilon_2 \in Ext_\sigma(\mathcal{AS}^A_{\mathcal{K}})$ and vice-versa (the corresponding extension can be found via repairs).*

Example 11 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be an inconsistent knowledge base such that:*

- $\mathcal{F} = \{a(m), b(m), c(m)\}$
- $\mathcal{R} = \{\forall x(a(x) \rightarrow d(x)\}$
- $\mathcal{N} = \{\forall x(a(x) \wedge b(x) \wedge c(x) \rightarrow \bot\}$

We deduce that $Repair(\mathcal{K}) = \{r_1, r_2, r_3\}$ where $r_1 = \{a(m), b(m)\}$, $r_2 = \{a(m), c(m)\}$ and $r_3 = \{c(m), b(m)\}$. Using Theorem 2.3.3, we deduce that for $\sigma \in \{preferred, stable\}$, $Ext_\sigma(\mathcal{AS}^M_{\mathcal{K}}) = \{Arg(r_1), Arg(r_2), Arg(r_3)\}$. Now, we instantiate $ASPIC^+$ using the mapping proposed in Definition 10. We obtain that $R_d = \{\emptyset \Rightarrow a(m), \emptyset \Rightarrow b(m), \emptyset \Rightarrow c(m), \emptyset \Rightarrow a(m) \wedge b(m), \emptyset \Rightarrow a(m) \wedge c(m), \emptyset \Rightarrow b(m) \wedge c(m)\}$ and $R_s = \{\forall x(a(x) \rightarrow d(x))\}$. Then, we construct the following arguments:

$A_1 : \emptyset \Rightarrow a(m)$ $\quad$ $A_4 : A_1 \rightarrow d(m)$
$A_2 : \emptyset \Rightarrow b(m)$
$A_3 : \emptyset \Rightarrow c(m)$
$B_1 : \emptyset \Rightarrow a(m) \wedge b(m)$ $\quad$ $B_4 : B_1 \rightarrow d(m)$
$B_2 : \emptyset \Rightarrow a(m) \wedge c(m)$ $\quad$ $B_5 : B_2 \rightarrow d(m)$
$B_3 : \emptyset \Rightarrow b(m) \wedge c(m)$

Following the definition of attacks and the definition of the contrariness function, we obtain the following attacks (see Figure 2.1).

We can easily verify that $\{Arg^A(R) | R \in Repair(\mathcal{K})\} = Ext_\sigma(\mathcal{AS}^A_{\mathcal{K}})$. For instance, the extension $Arg^A(r_1) = \{A_1, A_2, A_4, B_1, B_4\}$ is preferred and stable.

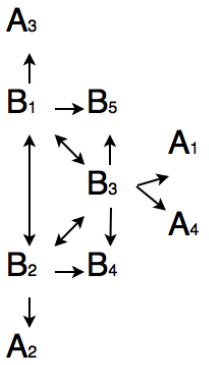

Figure 2.1 – Graph of attacks between arguments

2.4 Conclusion

In this chapter, we first introduced the theoretical background used in knowledge representation needed to comprehend this study (*Existential rules* and *Dung abstract argumentation framework*). The fundamental idea to remember was that in order to handle inconsistencies in a knowledge base, different methods are available such that *inconsistent tolerant semantics*, *Default Logics* and *argumentation*. We then showed two ways of instantiating argumentation frameworks for $Datalog^{\pm}$ following [17, 16] or [20]. In the case of the second instantiation, we introduced the first $ASPIC^{+}$instantiation for $Datalog^{\pm}$. We specified several definitions to formally define the argument structure and the attack relation. Facing the large number of definitions , the reader might wonder which ones are new. To answer this question, we recall here the new definitions based on their functions:

- Construction of arguments: Definition 5, Definition 8 and Definition 10.
- Attacks among arguments: Definition 7.

The main result of this chapter is that there is a semantics equivalence between the two instantiations, more precisely, we showed that for each preferred (resp. stable) extension of the first instantiation, there is a corresponding preferred (resp. stable) extension in the second instantiation. This result is interesting because it enables us to reuse all the tools already implementing the $ASPIC^{+}$argumentation framework with the knowledge that they wield the same results as the first instantiation. In the rest of this study, we will then work with both interchangeably. In the following chapter, we will introduce preferences in both frameworks because they play an important role in decision-making [19]. We will briefly explain how they appear in both frameworks.

3

Preferences And Logical Argumentation Frameworks

Preferences play an important role in our daily lives. They appear as soon as we are confronted we a choice, e.g. ”which movie should I watch tonight ?”, ”what kind of strawberries should I choose ?”. Among multiple possibilities, it is often necessary to choose one or more *alternatives* that are more appealing than the others. That is the reason we wanted to integrate preferences in the argumentation frameworks. The preference notion is a vast subject that has been thoroughly studied in different domains such as *Database Queries* or *Aggregation* [19]. In this chapter, we are focusing on the interests of preferences in the context of argumentation. All proofs of this chapter are available in Appendix E.

3.1 Preferences In Argumentation Frameworks

In this section, we will only explain the effect of preferences on the output of the argumentation frameworks and how it is computed. We will not answer the question of the *origin of the preference relation* but the interested reader may find some clues in [19].

3.1.1 Two Roles Of Preferences In Argumentation Frameworks

As explained in [5], preferences intervene with two roles in argumentation frameworks: in the *computation of extensions* (by deleting attacks for instance) and in the *refinement* of those solutions (i.e return only certain preferred extensions). The two roles are independent and obey distinct postulates [5].

Example 12 *Suppose that we have two arguments following the discussion between a child and an expert.*

- *Expert: Strawberries are accessory fruits because they are not berries. (a)*
- *Child: Strawberries are berries. (b)*

It is obvious that argument b attacks a. If we add the preference $a > b$, then the first method removes the attack and we obtain the extension $\{a, b\}$. Therefore, this method does not respect the conflict-freeness postulate[1] *whereas the second method does.*

Here, we will discuss two papers following the two distinct roles. In [16], preferences are introduced as a $\geq$ relation on facts in order to take into account the level of significance of the knowledge sources. Inspired by the work of [25] on globally, Pareto and completion optimal repairs, they defined the notion of locally optimal, Pareto optimal and globally optimal extensions. The second paper [20] sees preferences as a relation on defeasible rules to represent the probabilistic strength of each rule. This leads to a relation on arguments and deletion of some attacks.

3.1.2 Preferences In The Existing Instantiation for $Datalog^{\pm}$

Let us start by defining how preferences works in the instantiation proposed in Subsection 2.2.1. The preferences on facts refine the set of extensions into three sets (locally, Pareto and globally optimal) of extensions.

Definition 12 ***[cf [16]].*** *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N}, \geq)$ be a knowledge base with $\geq$ the relation preference on facts, and $\mathcal{AS}^M_{\mathcal{K}} = (\mathcal{A}, \mathcal{C})$ the corresponding argumentation framework. Then, if $A \geq B$ and $B \not\geq A$ then A is strictly preferred to B (denoted by $A > B$). Let σ be a semantics and ε an extension with respect to σ.*

- *ε is locally optimal (LO) iff $\nexists \phi \in Base(\varepsilon)$ and $\psi \in \mathcal{F} \backslash Base(\varepsilon)$ such that $Arg((Base(\varepsilon) \backslash \phi) \cup \psi)$ is conflict-free and $\psi > \phi$.*
- *ε is Pareto optimal (PO) iff $\nexists X \subseteq Base(\varepsilon)$ and $\psi \in \mathcal{F} \backslash Base(\varepsilon)$ such that $Arg((Base(\varepsilon) \backslash X) \cup \psi)$ is conflict-free and $\forall \phi \in X, \psi > \phi$.*
- *ε is globally optimal (GO) iff $\nexists X \subseteq Base(\varepsilon)$ and $Y \subseteq \mathcal{F}$ and $X \neq \emptyset$ such that $Arg((Base(\varepsilon) \backslash X) \cup Y)$ is conflict-free and $\forall \phi \in X, \exists \psi \in Y$ such that $\psi > \phi$.*

We denote by $Ext^{LO}_{\sigma}(\mathcal{AS}^M_{\mathcal{K}})$ (resp. $Ext^{PO}_{\sigma}(\mathcal{AS}^M_{\mathcal{K}})$, $Ext^{GO}_{\sigma}(\mathcal{AS}^M_{\mathcal{K}})$) the set of extensions locally (resp. Pareto, globally) optimal repairs under semantics σ.

The following example illustrates Definition 12 on the knowledge base of Example 6.

Example 13 *From the inconsistent knowledge base, we can compute the two repairs $r_1 = \{strawberry(Allstar)\}$ and $r_2 = \{basketball(Allstar)\}$. From that we conclude that there is two preferred extensions, $Ext_{preferred}(\mathcal{AS}^M_{\mathcal{K}}) = \{Arg(r_1), Arg(r_2)\}$. Now suppose that we get the preferences $strawberry(Allstar) > basketball(Allstar)$ then $Arg(r_1)$ is locally optimal whereas $Arg(r_2)$ is not.*

We now provide two examples showing that the sets of locally optimal, Pareto optimal and globally optimal extensions can be different from each other. The following example shows that the set of locally optimal extensions is different than the set of Pareto optimal and globally optimal extensions.

[1] This postulate specifies that each extension should be conflict-free (see Definition 1).

Example 14 *Let $\mathcal{AS}^M_{\mathcal{K}} = (\mathcal{A}, \mathcal{C})$ be an argumentation framework obtained from $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ where $\mathcal{F} = \{a(m), b(m), c(m), d(m)\}$, $\mathcal{R} = \emptyset$ and $\mathcal{N} = \{\forall x(a(x) \wedge c(x) \rightarrow \bot), \forall x(d(x) \wedge a(x) \rightarrow \bot), \forall x(c(x) \wedge b(x) \rightarrow \bot), \forall x(b(x) \wedge d(x) \rightarrow \bot)\}$. Then, we obtain two preferred extensions $E_1 = Arg(\{a(m), b(m)\})$ and $E_2 = Arg(\{c(m), d(m)\})$. Suppose now that we get the partial preferences $c(m) < a(m)$ and $d(m) < a(m)$. We get that $Ext^{LO}_{preferred}(\mathcal{AS}^M_{\mathcal{K}}) = \{E_1, E_2\}$ because we cannot exchange one element of each base with another element without creating a conflict. But E_2 is neither Pareto nor globally optimal because we can exchange $\{c(m), d(m)\}$ with $\{a(m)\}$. We conclude that $Ext^{PO}_{preferred}(\mathcal{AS}^M_{\mathcal{K}}) = Ext^{GO}_{preferred}(\mathcal{AS}^M_{\mathcal{K}}) = \{E_1\}$.*

Example 15 shows that the set of globally optimal extensions is different than the set of Pareto optimal and locally optimal extensions.

Example 15 *Let $\mathcal{AS}^M_{\mathcal{K}} = (\mathcal{A}, \mathcal{C})$ be an argumentation framework obtained from $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ where $\mathcal{F} = \{a(m), b(m), c(m), d(m)\}$, $\mathcal{R} = \emptyset$ and $\mathcal{N} = \{\forall x(a(x) \wedge d(x) \rightarrow \bot), \forall x(c(x) \wedge b(x) \rightarrow \bot)\}$. Let us consider the preferred extension $E = Arg(\{a(m), c(m)\})$ and the partial preferences $a(m) < b(m)$ and $c(m) < d(m)$. Then E is locally optimal and Pareto optimal because we cannot exchange one element of $Base(E)$ with another element because of conflicts or absence of preferences but E is not globally optimal.*

Please note that it was shown in [16] that $Ext^{GO}_{\sigma}(\mathcal{AS}^M_{\mathcal{K}}) \subseteq Ext^{PO}_{\sigma}(\mathcal{AS}^M_{\mathcal{K}}) \subseteq Ext^{LO}_{\sigma}(\mathcal{AS}^M_{\mathcal{K}})$. The authors also showed that the set of locally optimal (resp. Pareto optimal, globally optimal) extensions under preferred or stable semantics and the set of arguments built on locally optimal (resp. Pareto optimal, globally optimal) repairs coincide. The most important aspect of this work is that this preference-based argumentation framework also satisfies the *rationality postulates*.

3.1.3 Preferences In The Instantiation Using $ASPIC^+$For $Datalog^{\pm}$

Let us now study preferences in the instantiation introduced in Subsection 2.2.2. Preferences in $ASPIC^+$are seen as a $\preceq$ binary ordering on the set of all arguments. If $A \preceq B$ and $B \not\preceq A$ then B is strictly preferred to A (denoted by $A \prec B$). if $A \preceq B$ and $B \preceq A$ then we write that $A \approx B$. The rebutting and undermining of Definition 9 are slightly changed so that it incorporates the idea that argument a cannot attack an argument b that is stronger, this kind of attacks are called *critical attacks*[2]. Please note that the undercut remains unchanged by preferences. In [5], the author reminds that blindly removing all critical attacks may lead to conflicting extensions and consequently the framework may violate postulates (specially *conflict-freeness*[3]) in the case of an asymmetric attack relation.

Definition 13 ***[cf [20]].*** *Let a and b be arguments and $\preceq$ the binary ordering on the set of all arguments, we say that a successfully defeats b iff a undercuts, successfully undermines or successfully rebuts b, where:*

- *a successfully rebuts argument b if a rebuts b on b' and $a \not\prec b'$.*

[2]Let a,b be two arguments, a critical attack is an attack $(b, a) \in \mathcal{C}$ with $a > b$.

[3]Let $\mathcal{AS} = (\mathcal{A}, \mathcal{C})$ be an argumentation framework, $\mathcal{AS}$ respect the conflict-freeness postulates iff every extension is conflict-free w.r.t $\mathcal{C}$.

- *a successfully undermines argument b if a undermines b on ψ and $a \not\prec \psi$.*
- *a defeats b iff a undercuts or successfully rebuts or successfully undermines b.*

In order to define the $\preceq$ binary ordering on the set of all arguments, some ordering methods have been given by [20]. They define two argument preference orderings, called *weakest-link* and *last-link* ordering. The weakest-link principle compares the set of all the defeasible rules in the compared arguments whereas the last-link principle considers only the top defeasible rules. Here, we will only show how they defined the *last-link* ordering. We denote by $\geq_d$ the priority ordering on $\mathcal{R}_d$ (defeasible rules) and $\geq_p$ the one on $\mathcal{K}_p$ (ordinary premises). We also need to define an ordering $\trianglelefteq_s (s \in \{Elitist, Democratic\})$ on set of rules/premises because when we compare two arguments, we want to compare them on the set of rules/premises that these arguments are composed of. The *Elitist* way of defining $\trianglelefteq_s$ compares sets on their minimal elements and *Democratic* on their maximal elements. We assume that $\trianglelefteq_{Elitist}$ is used in the rest of this thesis.

Definition 14 ***[cf [20]].*** *Let Γ, Γ' be two finite set of rules/premises and $\geq$ a preordering on $\Gamma \cup \Gamma'$. Then $\trianglelefteq_s$ is defined as follows:*

(1) if $\Gamma = \emptyset$ then $\Gamma \not\trianglelefteq_s \Gamma'$. We shall say that Γ' is not preferred to Γ.

(2) else if $\Gamma' = \emptyset$ and $\Gamma \neq \emptyset$ then $\Gamma \trianglelefteq_s \Gamma'$. We shall say that Γ' is preferred to Γ.

(3) else if $s = Elitist$ then $\Gamma \trianglelefteq_s \Gamma'$ if $\exists x \in \Gamma$ s.t. $\forall y \in \Gamma', x \leq$ [4]y

(4) else if $s = Democratic$ then $\Gamma \trianglelefteq_s \Gamma'$ if $\forall x \in \Gamma, \exists y \in \Gamma', x \leq y$

In order to compare arguments, it is important to compare them on their certainty. One way of doing so is to compare them on the set of defeasible rules they are made of. The $LastDefRules : \mathcal{A} \rightarrow \mathcal{R}_d$ function returns the set of top defeasible rules of an argument.

Definition 15 (LastDefRules) ***[cf [20]].*** *Let A be an argument.*

- *$LastDefRules(A) = \emptyset$ if $DefRules(A) = \emptyset$.*
- *if $A = A_1, \ldots, A_m \Rightarrow c$, then $LastDefRules(A) = \{r = B \Rightarrow H |$ there exists a homomorphism σ from B to $X = Conc(A_1) \wedge Conc(A_2) \wedge \cdots \wedge Conc(A_m)$ and $c = \alpha(X, r, \sigma)\}$.*
  else $LastDefRules(A) = LastDefRules(A_1) \cup \cdots \cup LastDefRules(A_m)$.

Example 16 *Let $\mathcal{K}_p = \{strawberry(Allstar)\}$ be the set of ordinary premises, $\mathcal{R}_d = \{\forall x (strawberry(x) \Rightarrow fruit(x))\}$ the set of defeasible rules and $\mathcal{R}_s = \{\forall x (fruit(x) \rightarrow sweet(x))\}$ the set of strict rules. The argumentation framework contains the following arguments:*

A_1 : $\emptyset \Rightarrow strawberry(Allstar)$
A_2 : $A_1 \Rightarrow fruit(Allstar)$
A_3 : $A_2 \rightarrow sweet(Allstar)$

We deduce that $LastDefRules(A_3) = LastDefRules(A_2) = \{\forall x (strawberry(x) \Rightarrow fruit(x))\}$.

[4] This ordering corresponds to either $\leq_d$ or $\leq_p$ depending of the nature of x and y

Definition 16 (Last-link principle) ***[cf [20]].*** *Let A and B be two arguments. We say that $A \preceq B$ iff:*

- $LastDefRules(A) \trianglelefteq_s LastDefRules(B)$ *or*
- $LastDefRules(A) = LastDefRules(B) = \emptyset$ *and* $Prem(A) \trianglelefteq_s Prem(B)$.

Example 17 ***[cf [20]].*** *Let $\mathcal{K}_p = \{strawberry(Allstar), basketball(Allstar)\}$ be the ordinary premises, $\mathcal{R}_d = \{\forall x(strawberry(x) \Rightarrow_{r_1} fruit(x)), \forall x(fruit(x) \Rightarrow_{r_2} sweet(x)), \forall x (basketball(x) \Rightarrow_{r_3} sport(x))\}$, $sweet(Allstar) \in cf(sport(Allstar))$ and $sport(Allstar) \in cf(sweet(Allstar))$. We get the following arguments:*

$A_1 : \ \emptyset \Rightarrow strawberry(Allstar)$ $\qquad$ $B_1 : \ \emptyset \Rightarrow basketball(Allstar)$

$A_2 : \ A_1 \Rightarrow fruit(Allstar)$ $\qquad$ $B_2 : \ B_1 \Rightarrow sport(Allstar)$

$A_3 : \ A_2 \Rightarrow sweet(Allstar)$

Assume that $strawberry(Allstar) <_p basketball(Allstar)$ and $r_1 <_d r_2, r_1 <_d r_3, r_3 <_d r_2$. We get that $LastDefRules(A_3) = \{r_2\}$ and $LastDefRules(B_2) = \{r_3\}$. Since, $r_3 <_d r_2$ we have that $LastDefRules(B_2) \trianglelefteq_{Elitist} LastDefRules(A_3)$, hence $B_2 \prec A_3$. So A_3 strictly defeats B_2.

3.2 Preference Translation And Equivalence

In Section 2.3, we showed that there was an equivalence between extensions in the two instantiated frameworks under particular *acceptability semantics*. In this section, we will explain how facts-translated preferences in the $ASPIC^+$ instantiation defined in Section 3.1.3 affects the equivalence, i.e. the correspondence between the new set of extensions and the set of locally (resp. Pareto, globally) optimal extensions. This reasoning is depicted as follow (see Figure 3.1).

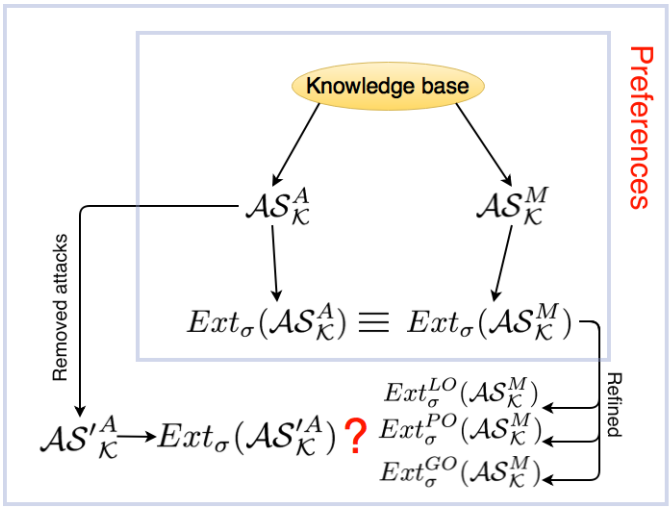

Figure 3.1 – Representation of open questions on equivalence results.

We will show that depending of how we translate preferences from facts into premises preferences, it affects greatly the preference relation on arguments.

3.2.1 Equivalence Results

Aiming toward a better understanding, let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base with a partial binary relation $\leq$ on elements of $\mathcal{F}$, we will distinguish between $ASPIC^+$instantiation $\mathcal{AS}^A_{\mathcal{K}}$ defined in Section 2.2.2 and the $ASPIC^+$instantiation extended with preferences $\mathcal{AS}'^A_{\mathcal{K}}$ defined in Section 3.1.3. First, we will recall some definitions introduced in Section 3.1.3 about how we import preferences in the $ASPIC^+$instantiation by constructing a partial relation on premises.

Definition 17 (Elitist ordering on sets of premises) *Let a and b be two arguments, we say that $Prem(a) \lhd_s Prem(b)$ iff there exists $\phi \in Prem(a)$ s.t. $\forall \psi \in Prem(b), \phi \leq \psi$.*

Definition 18 (Democratic ordering on sets of premises) *Let a and b be two arguments, we say that $Prem(a) \lhd_s Prem(b)$ iff $\forall \phi \in Prem(a), \exists \psi \in Prem(b), \phi \leq \psi$.*

In order to complete the definition of the preference-based argumentation framework, we need to further detail how conjunction of facts are compared (because in the instantiation, premises are conjunctions of facts). We define two possible relations on conjunctions of facts.

Definition 19 (Elitist relation on conjunctions of facts) *Let C_1 and C_2 be two $ASPIC^+$premises. We say that $C_1 \leq C_2$ iff there exists a fact c_1 of $Elimination(C_1)$ such that for each fact c_2 of $Elimination(C_2)$, $c_1 \leq c_2$.*

Definition 20 (Democratic relation on conjunctions of facts) *Let C_1 and C_2 be two $ASPIC^+$premises. We say that $C_1 \leq C_2$ iff for all facts c_1 of $Elimination(C_1)$, there exists a fact c_2 of $Elimination(C_2)$, $c_1 \leq c_2$.*

From now on, we will consider that we are using Definition 17 and Definition 19. The first result of this paper is to show that stable extensions obtained in $\mathcal{AS}'^A_{\mathcal{K}}$ are contained in the set of stable extensions of $\mathcal{AS}^A_{\mathcal{K}}$.

The first lemma proves that the stable extensions of $\mathcal{AS}'^A_{\mathcal{K}}$ are closed under Sub arguments.

Lemma 3.2.1 *Let ε be a stable extension of $\mathcal{AS}'^A_{\mathcal{K}}$ and $a \in \varepsilon$ then $Sub(a) \subseteq \varepsilon$.*

The following lemma shows that if there are two simple arguments a and b such that a includes all the b's elements, then b is in the same extensions as a.

Lemma 3.2.2 *Let a, b be two arguments s.t. $a : \emptyset \Rightarrow c_1$ and $b : \emptyset \Rightarrow c_2$ with $Elimination(c_1) \subset Elimination(c_2)$ and ε be a stable extension of $\mathcal{AS}'^A_{\mathcal{K}}$. If $b \in \varepsilon$ then $a \in \varepsilon$.*

The next lemma follows from the two previous lemma and states that for every elements of a premise of an argument contained in a stable extension of $\mathcal{AS}'^A_{\mathcal{K}}$, there exists a corresponding basic argument included in the extension.

Lemma 3.2.3 *Let ε be a stable extension of $\mathcal{AS}'^A_{\mathcal{K}}$ and $a \in \varepsilon$. If $c \in Prem(a)$ then $\forall \phi \in Elimination(c)$ we have $(\emptyset \Rightarrow \phi) \in \varepsilon$.*

The following theorem shows that stable extensions of $\mathcal{AS}'^A_{\mathcal{K}}$ are included in the set of stable extensions of $\mathcal{AS}^A_{\mathcal{K}}$. It is an important result as it implies that adding preferences in the $ASPIC^+$argumentation framework cannot create extensions.

Theorem 3.2.4 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{AS}^A_\mathcal{K}$ the $ASPIC^+$ instantiation and $\mathcal{AS}'^A_\mathcal{K}$ be the corresponding $ASPIC^+$ instantiation with preferences. Then:*

$$Ext_{stable}(\mathcal{AS}'^A_\mathcal{K}) \subseteq Ext_{stable}(\mathcal{AS}^A_\mathcal{K})$$

Please note that as expressed by Example 18, one can see that we don't have the equality, i.e. $Ext_{stable}(\mathcal{AS}^A_\mathcal{K}) \not\subseteq Ext_{stable}(\mathcal{AS}'^A_\mathcal{K})$.

Example 18 *Let us consider a knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ where $\mathcal{F} = \{a(m), b(m)\}$, $\mathcal{R} = \emptyset$ and $\mathcal{N} = \{\forall x(a(x) \wedge b(x) \rightarrow \bot)\}$. We obtain two $ASPIC^+$ arguments:*

- $A_1 : \emptyset \Rightarrow a(m)$
- $A_2 : \emptyset \Rightarrow b(m)$

It is clear that A_1 and A_2 attack each other and that there are two stable extensions in $Ext_{stable}(\mathcal{AS}^A_\mathcal{K}) = \{\{A_1\}, \{A_2\}\}$. But if we add the preferences $a(m) > b(m)$ then we get $A_2 \prec A_1$ and the only resulting stable extension of $Ext_{stable}(\mathcal{AS}'^A_\mathcal{K})$ is $\{A_1\}$.

We will now state the main result of this paper, i.e. the equivalence between stable extensions of $\mathcal{AS}'^A_\mathcal{K}$ and the set $\{Arg^A(R) | R$ is Pareto optimal$\}$.

Theorem 3.2.5 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{AS}^A_\mathcal{K}$ be the corresponding $ASPIC^+$ instantiation. Then:*

$$Ext_{stable}(\mathcal{AS}'^A_\mathcal{K}) = \{Arg^A(R) | R \textit{ is a Pareto optimal repair}\}$$

We recall in the figure below all the inclusions deduced from the structured argumentation framework using Definition 17 and Definition 19.

$$\begin{array}{rcl} & \supsetneq & \{Arg^A(R) | R \text{ is a globally optimal repair}\} \\ & & \cap| \\ Ext(\mathcal{AS}'_\mathcal{K}) & = & \{Arg^A(R) | R \text{ is a Pareto optimal repair}\} \\ \cap| & & \cap| \\ & \subsetneq & \{Arg^A(R) | R \text{ is a locally optimal repair}\} \\ & & \cap| \\ Ext(\mathcal{AS}_\mathcal{K}) & = & \{Arg^A(R) | R \in Repair(\mathcal{K})\} \end{array}$$

Figure 3.2 – Representation of different inclusions proved in this section.

3.2.2 Non-Equivalence Results

Now, we will try to investigate the preferences in $ASPIC^+$ using different combinations of Definition 17, 18, 19 and 20. First, we will show that using either Definition 18 and 20, 17 and 20 or either 18 and 19 will result in no equality between the set of extensions of $\mathcal{AS}'^A_\mathcal{K}$ and the set of extensions corresponding to locally (resp. Pareto, globally) optimal repairs. The following theorem is straightforward and states that if the set of stable extensions of $\mathcal{AS}'^A_\mathcal{K}$ is not included in the set of extensions that can be constructed on locally optimal repairs, then it

cannot be included in the set of extensions constructed on Pareto optimal repairs (resp. globally optimal repairs).

Theorem 3.2.6 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{AS}'^{A}_{\mathcal{K}}$ be the corresponding $ASPIC^{+}$ instantiation with preferences. Then if $Ext_{stable}(\mathcal{AS}'^{A}_{\mathcal{K}}) \not\subseteq \{Arg^{A}(R)|$ R is a locally optimal repair$\}$ then:*

- $Ext_{stable}(\mathcal{AS}'^{A}_{\mathcal{K}}) \not\subseteq \{Arg^{A}(R)|$ R is a globally optimal repair$\}$
- $Ext_{stable}(\mathcal{AS}'^{A}_{\mathcal{K}}) \not\subseteq \{Arg^{A}(R)|R$ is a Pareto optimal repair$\}$.

In the rest of this section, we will prove that for any combinations of definitions mentioned above, we can apply Theorem 3.2.6. We first show that using Definition 18 and Definition 20, the set of stable extensions of $\mathcal{AS}'^{A}_{\mathcal{K}}$ is not included in the set of extensions that can be constructed on locally optimal repairs. We can thus apply Theorem 3.2.6.

Proposition 3.2.7 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{AS}'^{A}_{\mathcal{K}}$ be the corresponding $ASPIC^{+}$ instantiation with preferences using Definition 18 and Definition 20. Then:*

$$Ext_{stable}(\mathcal{AS}'^{A}_{\mathcal{K}}) \not\subseteq \{Arg^{A}(R)|R \text{ is a locally optimal repair}\}$$

Similarly, using Definition 18 and Definition 19, the set of stable extensions of $\mathcal{AS}'^{A}_{\mathcal{K}}$ is not included in the set of extensions that can be constructed on locally optimal repairs. We can thus apply Theorem 3.2.6.

Proposition 3.2.8 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{AS}'^{A}_{\mathcal{K}}$ be the corresponding $ASPIC^{+}$ instantiation with preferences using Definition 18 and Definition 19. Then:*

$$Ext_{stable}(\mathcal{AS}'^{A}_{\mathcal{K}}) \not\subseteq \{Arg^{A}(R)|R \text{ is a locally optimal repair}\}$$

Finally, using Definition 17 and Definition 20, the set of stable extensions of $\mathcal{AS}'^{A}_{\mathcal{K}}$ is also not included in the set of extensions that can be constructed on locally optimal repairs. Here again, we can apply Theorem 3.2.6.

Proposition 3.2.9 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{AS}'^{A}_{\mathcal{K}}$ be the corresponding $ASPIC^{+}$ instantiation with preferences using Definition 17 and Definition 20. Then:*

$$Ext_{stable}(\mathcal{AS}'^{A}_{\mathcal{K}}) \not\subseteq \{Arg^{A}(R)|R \text{ is a locally optimal repair}\}$$

It is interesting to note that in the specific case of using Definition 18 and Definition 20, the set of arguments that can be obtained from a globally optimal repair is contained in the set of stable extensions of an $ASPIC^{+}$instantiation extended with preferences.

Theorem 3.2.10 *Let $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$ be a knowledge base, $\mathcal{AS}'^{A}_{\mathcal{K}}$ be the corresponding $ASPIC^{+}$ instantiation with preferences using Definition 18 and 20. Then:*

$$Ext_{stable}(\mathcal{AS}'^{A}_{\mathcal{K}}) \supseteq \{Arg^{A}(R)|R \text{ is a globally optimal repair}\}$$

3.3 Conclusion

In this chapter, we described how preferences are taken into account in the two instantiations. On the one hand, the first instantiation uses preferences on facts in order to refines the set of extensions produced into three set of extensions: locally, Pareto and globally optimal extensions sets. On the other hand, the second instantiation builds a preference relation on arguments in order to remove attacks, thus modifying the set of extensions produced. Since the preference relation on arguments in the second instantiation is derived from the ordering on defeasible rules and ordinary premises (which are conjunction of facts), we defined two ways of comparing conjunctions of facts:

- Elitist relation on conjunctions of facts.
- Democratic relation on conjunctions of facts.

It may be difficult to expect the previous equivalence on both argumentation frameworks to still hold after applying preferences. However, we showed that depending on how we translate the preferences from facts to preferences on premises and how the preference relation on premises transfers to the preference relation on arguments, some equivalence results can hold. In the case of using Definition 17 and Definition 19, each Pareto extension of the first instantiation has a corresponding stable extension in the second instantiation extended with preferences. We also showed that in three other combinations, the previous result does not hold.

In the next chapter, we will introduce a real use-case concerning the end-of-life of packagings. We will also verify the equivalence result on both frameworks extended with preferences in this use-case.

4

Implementation

In this chapter, we will describe the use-case we obtained from several meetings with experts concerning the end-of-life of packagings. We will also show that the preferences module we added in the existing web application works according to the theory. Last but not least, we will use another project called "ASPIC project" to confirm the equivalence result.

4.1 Use-Case

The use-case presents the arguments given by several stakeholders (consumers, experts, researchers, etc.) regarding the end-of-life possibilities for packagings (Anaerobic digestion, Incineration, Landfill, Composting, Recycling). We formalized the text arguments in Appendix F into concepts (Appendix G) and rules (Appendix H.1). Note that we chose to implement end-of-life possibilities as elements of the knowledge base as it would be easier to characterize extensions by the elements of their bases. Furthermore, the elements of $\mathcal{F}$ are non-exclusive by default so we added some negative constraints (Appendix H.2). The first part of this section will deal with the new preferences module of EcoBioCap and the results obtained for this use-case. Then, we will validate the equivalence result with the Advanced Computation Laboratory's ASPIC project.

4.1.1 EcoBioCap

Within the framework of the European project EcoBioCap [26, 27, 28], a Decision Support System (DSS) has been implemented as a java GXT/GWT web application accessible online at `http://pfl.grignon.inra.fr/EcoBioCapProduction/` (although the access is restricted). From an argumentation point of view, despite all the available softwares in the field of argumentation, there were no argumentation software implementing an argumentation process from argument expression to extensions computation. Not only does this software implement the entire process but it also provides several GUI for visualization purposes. It is composed of four steps: formalizing text arguments, processing arguments (according to $ASPIC^+$ argumentation framework), computing extensions[1] and enriching the multi-criteria flexible querying. In order to use this tool for decision-making even when the engine returns more than one extension, we introduced a preferences module.

[1] Please note that it does not contain undercutting attacks.

4.1.2 Preferences Module

In order to add preferences to the existing software, one can either choose to implement $ASPIC^+$ preferences from an ordering on premises and defeasible rules as described in Subsection 3.1.3 or try to import the preferences of Subsection 3.1.2. We chose to implement the latter. We integrated a simple and intuitive interface for inputting preferences enabling users to clearly visualize the preferences implied and the possible incoherences (see Figure 4.1).

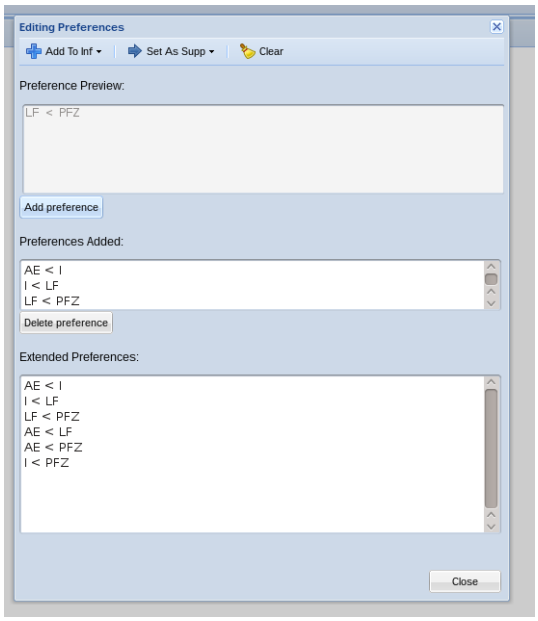

Figure 4.1 – Preference interface in EcobioCap.

We inputted elements of $\mathcal{F}$ as choices and after calculation, we got five extensions (see Figure 4.2):

- $\varepsilon_1 = \{AE(m), PGS(m), PEV(m), HIP(m), VPL(m), DPR(m), LCD(m)\}$
- $\varepsilon_2 = \{I(m), DXP(m), PEN(m)\}$
- $\varepsilon_3 = \{LF(m), FNL(m), LCT(m), LTE(m)\}$
- $\varepsilon_4 = \{C(m), PFZ(m), PEV(m), HIP(m), VPL(m), DPR(m), HEP(m), APP(m), LCD(m)\}$
- $\varepsilon_5 = \{R(m), LCA(m), ETX(m)\}$.

If we add the following partial preferences: $PEN(m) < AE(m), I(m) < AE(m), PEN(m) < LF(m), I(m) < LF(m), PEN(m) < PGS(m), I(m) < PGS(m), AE(m) < C(m), LF(m) < C(m), PGS(m) < C(m), C(m) < PFZ(m), C(m) < R(m)$, the new preferences module is able to sort the preferred extensions according to three semantics: locally optimal, Pareto optimal and globally optimal. For these preferences, we get that ε_3 is not locally optimal, ε_2 and ε_1 are locally optimal and $\varepsilon_5, \varepsilon_4$ are globally optimal. Please find the results in Figure 4.3.

Following this result, we can say that according to the preferences stated, the two preferred end-of-life possibilities are ”Recycling” and ”Compostable”.

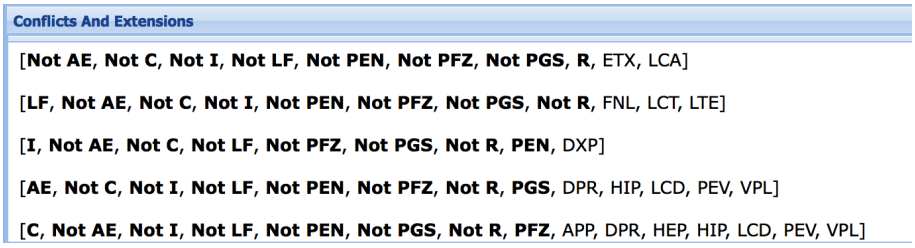

Figure 4.2 – Set of preferred extensions calculated by the EcoBioCap application. Please notice that bold elements represent choices.

Not locally optimal	Locally optimal	Pareto optimal	Globally optimal
	ε_5	ε_5	ε_5
	ε_4	ε_4	ε_4
	ε_1		
	ε_2		
ε_3			

Figure 4.3 – Overview of the results after application of the preferences.

4.2 Practical Validation

In this section, we validate the equivalence result on both frameworks extended with preferences by using the ACL's ASPIC argumentation engine. This engine can process simple queries according to semantics (we used preferred credulous[2]) and it accepts different settings (Weakest-Link, Last-Link, etc.). According to the equivalence result, each element of a Pareto extension should answer "Yes".

4.2.1 Using ASPIC Argumentation Engine

Aspic Inference Engine is an argumentation engine implemented as a Java application. The input is composed of facts, beliefs, strict rules and defeasible rules (with names) and the output is an answer to a query. The pertinence measures the degree of confidence in a belief or a rule. A pertinence of 1 represents a fact or a strict rule.

4.2.2 Request Answering

In order to give an answer to a request following a specific semantics, the software uses a method similar to the ones in [30]. The method implements two protagonists: PRO and CON that are confronting each other. CON provides arguments that attack arguments of PRO using rules and substitutions whereas PRO tries to defend itself from these attacks. If PRO wins, it means that the query is admissible, otherwise, it is not (see Figure 4.4 for a visual explanation of this process).

Moreover, one may notice that the pertinence of complex arguments, i.e. arguments of the form $A : A_1, \ldots, A_n \Rightarrow c$ (resp. $A : A_1, \ldots, A_n \rightarrow c$) differs depending of the valuation used:

[2] An argument is accepted under preferred credulous semantics iff it belongs to at least one preferred extension.

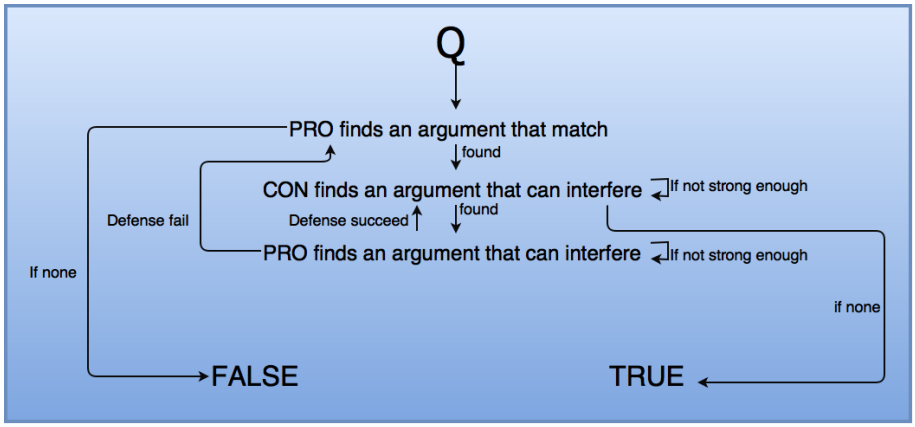

Figure 4.4 – Simplified view of the request answering system.

- Weakest-link: the pertinence of the argument A is equal to $min(w(A_1), \ldots, w(A_n), w_r)$
- Last-link: the pertinence of the argument A is equal to w_r if $w_r < 1$ or else it is equal to $min(w(A_1), \ldots, w(A_n))$

where $w(A_i), 1 \leq i \leq n$ denotes the pertinence of the argument A_i and w_r denotes the pertinence of the defeasible rule (resp. strict rule) applied.

Another problem we encountered was that although the query works well for single atom queries, it is ill-defined for compound queries as showed in Example 19. Please note that the Ruby Argumentation Engine Demo[3] simply returns "Not accepted" for every compound queries.

Example 19 *Let us suppose that the input is composed of three beliefs $a(m), b(m), c(m)$ of pertinence 0.5 and of three strict rules ($\forall x(b(x) \wedge c(x) \rightarrow \neg a(x))$, $\forall x(a(x) \wedge c(x) \rightarrow \neg b(x))$, $\forall x(a(x) \wedge b(x) \rightarrow \neg c(x))$). We get six arguments:*

$A_1 : \emptyset \Rightarrow a(m)$ $\quad A_4 : A_2, A_3 \rightarrow \neg a(m)$
$A_2 : \emptyset \Rightarrow b(m)$ $\quad A_5 : A_1, A_3 \rightarrow \neg b(m)$
$A_3 : \emptyset \Rightarrow c(m)$ $\quad A_6 : A_1, A_2 \rightarrow \neg c(m)$

If we use the weakest-link valuation, we get that A_4, A_5 and A_6 have a pertinence of 0.5. Then, if we give the engine the query "$a(m), b(m), c(m)$", the engine will try to defeat each of the components. For instance, CON will first try to defeat $a(m)$ and put forward A_4 as an attacker, however since A_4 is as pertinent as A_1, A_1 succeeds in defending itself against A_4. The same happens for $b(m)$ and $c(m)$. Finally, the engine concludes that the query is admissible.

However, if we follow the definition of attacks under preferences of [20] (and the Democratic/Elitist ways), we get the graph of attacks of Figure 4.5. We can clearly see that there are three preferred extensions: $\{A_1, A_2, A_5\}, \{A_1, A_3, A_6\}$ and $\{A_2, A_3, A_4\}$. It is evident that there is no preferred extension that can answer "$a(m), b(m), c(m)$".

4.2.3 The Model In Aspic

In order to implement the model, we have to encode elements of $\mathcal{F}$ as beliefs with pertinence according to the previous preferences (see Figure 4.6). Please note that the application does

[3] Available at `http://aspic.cossac.org/ArgumentationSystem/`

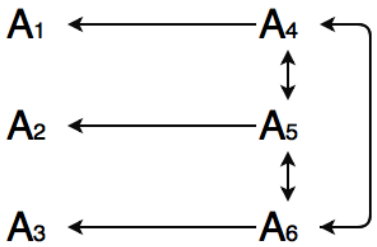

Figure 4.5 – Graph of attacks between arguments in the $ASPIC^+$argumentation framework with preferences.

not allow to encode partial preferences. However, it is interesting to note that in $ASPIC^+$, preferences on ordinary premises are not always complete. Please refer to [20] for an example.

Beliefs	Pertinence
ae(m)	0.5
c(m)	0.51
i(m)	0.4
lf(m)	0.5
pfz(m)	0.6
pgs(m)	0.5
r(m)	0.6
pen(m)	0.4

Figure 4.6 – Beliefs and their pertinences given as input. Please note that this application only accepts lowercase letters for predicate names.

Remark It should be noted that writing that $pev(m)$ belongs to ε_1 in Figure 4.7 is an abuse of notation. In fact, it means that the argument $A_1 \rightarrow pev(m)$ belongs to ε_1 with $A_1 : \emptyset \Rightarrow ae(m)$. Similarly, writing that $pev(m)$ belongs to ε_4 means that the argument $A_2 \rightarrow pev(m)$ belongs to ε_4 with $A_2 : \emptyset \Rightarrow c(m)$.

Extension	Base	Exclusive Arguments	Arguments Shared
ε_1	$ae(m), pgs(m)$		$pev(m), hip(m), vpl(m), dpr(m), lcd(m)$
ε_2	$i(m), pen(m)$	$dxp(m)$	
ε_3	$lf(m)$	$fnl(m), lct(m), lte(m)$	
ε_4	$c(m), pfz(m)$	$hep(m), app(m)$	$pev(m), hip(m), vpl(m), dpr(m), lcd(m)$
ε_5	$r(m)$	$lca(m), etx(m)$	

Figure 4.7 – Representation of extensions.

Figure 4.7 and 4.8 show that elements of $\mathcal{F}$ that are included in the base of a Pareto extension (resp. not Pareto extension) return Yes (resp. No) when asked as a query. We can also notice that the engine returns a "Yes/No" answer in the case of arguments shared by Pareto and not Pareto extensions because the engine was able to create two separate proofs. We showed in

Query	Answer (Yes/No)
ae(m)	No
i(m)	No
lf(m)	No
c(m)	Yes
r(m)	Yes
pev(m)	Yes/No
dxp(m)	No
hip(m)	Yes/No
hep(m)	Yes
app(m)	Yes
lca(m)	Yes
lcd(m)	Yes/No
etx(m)	Yes
fnl(m)	No
vpl(m)	Yes/No
lct(m)	No
dpr(m)	Yes/No
pen(m)	No
pgs(m)	No
pfz(m)	Yes
lte(m)	No

Figure 4.8 – Results of the different queries.

Theorem 3.2.5 that Pareto extensions coincide with $ASPIC^+$stable extensions, given specific settings (elitist preordering, strict rules, contrariness function, etc.). The results here do confirm the equivalence result since elements of Pareto extensions (see Figure 4.3) return a Yes answer as displayed in Figure 4.8.

4.3 Conclusion

In this chapter, we described a real life use-case we obtained from several meetings with experts concerning the end-of-life of packagings. Then, we showed that the preferences module we added in the existing EcoBioCap web application worked well on this use-case. We also described the querying method used in the "ASPIC project" and found the flaws it contained. Finally, we confirmed the equivalence result proved in the previous chapter by verifying that elements in Pareto extensions were contained in stable extensions.

5

Conclusion

The study includes the theoretical background as well as the latest state-of-art theories about structured argumentation framework using $Datalog^{\pm}$. We described several ways to handle inconsistent knowledge bases (repairs, inconsistent tolerant semantics, etc.) and introduced two different instantiations (one existing and one new) for existential rules.

The contribution of this study is first, in Chapter 2, a new $ASPIC^{+}$instantiation for $Datalog^{\pm}$ with all the necessary settings (contrariness function, arguments structure, etc.) and some equivalence results on extensions of both frameworks. Namely, we showed that for each extension under preferred or stable semantics in the first instantiation, there is a corresponding extension under the same semantics in the new $ASPIC^{+}$instantiation.

In Chapter 3, we analyzed how preferences on facts are handled by both systems. First, we recalled how preferences refine the set of extensions in the first instantiation. Secondly, we introduced several new definitions needed to instantiate the $ASPIC^{+}$framework with preferences. Then, we proved a correspondance between Pareto extensions and $ASPIC^{+}$stable extensions when using specific definitions. Lastly, we used multiple counter-examples to show that the set of stable extensions in the second instantiation (with preferences) is not included in the set of Pareto extensions in all other cases.

In the last chapter, we described a real life use-case we obtained from several meetings with experts concerning the end-of-life of packagings. Then, we showed that the preferences module (interface/database/computation) we added in the existing EcoBioCap web application worked according to the theory on this use-case. We also described the querying method used in the "ASPIC project" and found the flaws it contained. Finally, we confirmed the equivalence result proved in the previous chapter by using these two applications.

Please note that Definition 7, Definition 8, Definition 10, Definition 11, Theorem 2.3.2, Theorem 2.3.3, Theorem 2.3.4, Definition 19, Definition 20, Theorem 3.2.4, Theorem 3.2.5, Theorem 3.2.6, Proposition 3.2.7, Proposition 3.2.8, Proposition 3.2.9, Theorem 3.2.10 and the corresponding proofs are new results obtained thoughtout the course of this internship. They allowed us to put in context the state of the art and the semantics equivalences.

The work proposed in this study opens new promising research avenues:

- We proposed an instantiation using $ASPIC^{+}$for $Datalog^{\pm}$ with some particular settings. It would be interesting to find a new mapping that not only constructs ordinary premises and strict rules (as it is now) from a knowledge base but also creates defeasible rules and

associates formulas to defeasible rules. This would enable us to enjoy the full power of the $ASPIC^+$attack (undercutting, undermining, rebutting). Once this new mapping is fixed, it would be interesting to find new equivalence results on this instantiation.

- Another interesting notion are *Ranking semantics* introduced in [2]. Ranking semantics are functions that transforms any argumentation framework into a ranking (total and transitive binary relation) on arguments. To put it in a nutshell, they rank arguments on their acceptability. Using these semantics with new algorithms may prove to be a more intuitive alternative to preferences in the area of decision-making.

- We also noted that although we introduced a method to refine the set of extensions into three set of extensions (locally, Pareto and globally optimal) according to preferences on facts, it appears that in the area of decision-making, a preference relation on the facts that are generated may be more useful because preferences are often stated on the effects of decisions rather than on the decisions themselves. In that prospect, we thought about importing all facts that can be generated into the knowledge base and converting rules into negative constraints. By doing so, we could use the same theory and have preferences on all facts.

Index

Bibliography

[1] L. Amgoud. Five weaknesses of ASPIC +. In *Advances in Computational Intelligence - 14th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU 2012, Catania, Italy, July 9-13, 2012, Proceedings, Part III*, pages 122–131, 2012.

[2] L. Amgoud and J. Ben-Naim. Ranking-based semantics for argumentation frameworks. pages 134–147, 2013.

[3] L. Amgoud and P. Besnard. Bridging the gap between abstract argumentation systems and logic. In *Scalable Uncertainty Management, Third International Conference, SUM 2009, Washington, DC, USA, September 28-30, 2009. Proceedings*, pages 12–27, 2009.

[4] L. Amgoud and H. Prade. Explaining qualitative decision under uncertainty by argumentation. In *Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA*, pages 219–224, 2006.

[5] L. Amgoud and S. Vesic. Two roles of preferences in argumentation frameworks. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty - 11th European Conference, ECSQARU 2011, Belfast, UK, June 29-July 1, 2011. Proceedings*, pages 86–97, 2011.

[6] A. Arioua, N. Tamani, M. Croitoru, and P. Buche. Query failure explanation in inconsistent knowledge bases: a dialogical approach. In *Research and Development in Intelligent Systems XXXI*, pages 119–133. Springer International Publishing, 2014.

[7] F. Baader, S. Brandt, and C. Lutz. Pushing the el envelope. In *Proc. of IJCAI 2005*, 2005.

[8] J.-F. Baget, F. Garreau, M.-L. Mugnier, and S. Rocher. Revisiting Chase Termination for Existential Rules and their Extension to Nonmonotonic Negation. *ArXiv e-prints*, May 2014.

[9] J.-F. Baget, M. Leclère, M.-L. Mugnier, and E. Salvat. On rules with existential variables: Walking the decidability line. *Artificial Intelligence*, 175(10):1620 – 1654, 2011.

[10] J.-F. Baget, M.-L. Mugnier, S. Rudolph, and M. Thomazo. Walking the complexity lines for generalized guarded existential rules. In *Proc. of IJCAI'11*, pages 712–717, 2011.

[11] F. Bex, H. Prakken, and C. Reed. A formal analysis of the AIF in terms of the ASPIC framework. In *Computational Models of Argument: Proceedings of COMMA 2010, Desenzano del Garda, Italy, September 8-10, 2010.*, pages 99–110, 2010.

[12] A. Calì, G. Gottlob, and T. Lukasiewicz. Datalog$+/-$: a unified approach to ontologies and integrity constraints. In *Proc of ICDT'09*, pages 14–30. ACM, 2009.

[13] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The dl-lite family. *J. Autom. Reasoning*, 39(3):385–429, 2007.

[14] M. Chein and M. Mugnier. *Graph-based Knowledge Representation - Computational Foundations of Conceptual Graphs*. Advanced Information and Knowledge Processing. Springer, 2009.

[15] M. Croitoru, R. Thomopoulos, and N. Tamani. A practical application of argumentation in french agrifood chains. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems - 15th International Conference, IPMU 2014, Montpellier, France, July 15-19, 2014, Proceedings, Part I*, pages 56–66, 2014.

[16] M. Croitoru, R. Thomopoulos, and S. Vesic. Introducing preference-based argumentation to inconsistent ontological knowledge bases. In *PRIMA 2015: Principles and Practice of Multi-Agent Systems - 18th International Conference, Bertinoro, Italy, October 26-30, 2015, Proceedings*, pages 594–602, 2015.

[17] M. Croitoru and S. Vesic. What can argumentation do for inconsistent ontology query answering? In *Scalable Uncertainty Management - 7th International Conference, SUM 2013, Washington, DC, USA, September 16-18, 2013. Proceedings*, pages 15–29, 2013.

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

[19] S. Kaci. *Working with Preferences: Less Is More*. Cognitive Technologies. Springer, 2011.

[20] S. Modgil and H. Prakken. The *ASPIC*$^+$ framework for structured argumentation: a tutorial. *Argument & Computation*, 5(1):31–62, 2014.

[21] M. Mugnier. Ontological query answering with existential rules. In *Web Reasoning and Rule Systems - 5th International Conference, RR 2011, Galway, Ireland, August 29-30, 2011. Proceedings*, pages 2–23, 2011.

[22] H. Prakken. Formalising a legal opinion on a legislative proposal in the aspic$^+$ framework. In *Legal Knowledge and Information Systems - JURIX 2012: The Twenty-Fifth Annual Conference, University of Amsterdam, The Netherlands, 17-19 December 2012*, pages 119–128, 2012.

[23] H. Prakken and S. Modgil. Clarifying some misconceptions on the aspic$^+$ framework. In *Computational Models of Argument - Proceedings of COMMA 2012, Vienna, Austria, September 10-12, 2012*, pages 442–453, 2012.

[24] R. Reiter. A logic for default reasoning. *Artif. Intell.*, 13(1-2):81–132, 1980.

[25] S. Staworko, J. Chomicki, and J. Marcinkowski. Prioritized repairing and consistent query answering in relational databases. *Ann. Math. Artif. Intell.*, 64(2-3):209–246, 2012.

[26] N. Tamani, P. Mosse, M. Croitoru, P. Buche, and V. Guillard. A food packaging use case for argumentation. In *Metadata and Semantics Research - 8th Research Conference, MTSR 2014, Karlsruhe, Germany, November 27-29, 2014. Proceedings*, pages 344–358, 2014.

[27] N. Tamani, P. Mosse, M. Croitoru, P. Buche, V. Guillard, C. Guillaume, and N. Gontard. Eco-efficient packaging material selection for fresh produce: Industrial session. In *Graph-Based Representation and Reasoning - 21st International Conference on Conceptual Structures, ICCS 2014, Iaşi, Romania, July 27-30, 2014, Proceedings*, pages 305–310, 2014.

[28] N. Tamani, P. Mosse, M. Croitoru, P. Buche, V. Guillard, C. Guillaume, and N. Gontard. An argumentation system for eco-efficient packaging material selection. *Computers and Electronics in Agriculture*, 113:174 – 192, 2015.

[29] B. van Gijzel and H. Prakken. Relating carneades with abstract argumentation via the aspic$^+$ framework for structured argumentation. *Argument & Computation*, 3(1):21–47, 2012.

[30] G. Vreeswijk and H. Prakken. Credulous and sceptical argument games for preferred semantics. In *Logics in Artificial Intelligence, European Workshop, JELIA 2000 Malaga, Spain, September 29 - October 2, 2000, Proceedings*, pages 239–253, 2000.